ELSEVIER

Theoretical Computer Science 200 (1998) 1–44

Theoretical Computer Science

Fundamental Study

Nondeterministic operations on finite relational structures [1]

Klaus Barthelmann*

Institut für Informatik, Johannes Gutenberg-Universität Mainz, D-55099 Mainz, Germany

Received May 1997; revised October 1997
Communicated by M. Nivat

Abstract

This article builds on a tutorial introduction to universal algebra for language theory (Courcelle, Theoret. Comput. Sci. 163 (1996) 1–54) and extends it in two directions. First, nondeterministic operations are considered, i.e., operations which give a set of results instead of a single one. Most of their properties concerning recognizability and equational definability carry over from the ordinary case with minor modifications. Second, inductive sets of evaluations are studied in greater detail. It seems that they are handled most naturally in the framework presented here. We consider the analogues of top-down and bottom-up tree transducers. Again, most of their closure properties are maintained. We relate them to the logical theory of finite relational structures and hypergraphs. Our central theorem says that every transduction which is definable in counting monadic second-order logic is a bottom-up transduction on term representations. Finally, several examples indicate that one cannot hope for decidable logical theories in the presence of unbounded nondeterminism.

Keywords: Formal languages (recognizable and context-free sets, transducers); Logic; Hypergraphs

Contents

[1] This work was partially supported by the European Human Capital and Mobility program *Combinatorial Methods on Graphs for Distributed Computations* and the university Bordeaux I.

* E-mail: barthel@informatik.uni-mainz.de.

0304-3975/98/$19.00
PII S0304-3975(97)00263-6

1. Introduction

Notions of recognizable and context-free sets in general algebras were introduced in [26]. The idea was that an element of any algebra can be represented as a term over a suitable signature and that a tree automaton can run on this representation. If the outcome depends only on the value of the term then it makes sense to say that the automaton either accepts or rejects the element. An automaton is thus essentially a homomorphism into a finite algebra. A definition of context-free sets which can be generalized is that as a least solution of a recursive system of equations or, equivalently, as the set of normal forms of a ground term rewriting system.

These notions have been applied to graph rewriting systems to define properly the concepts of recognizable or context-free sets of graphs [4, 3, 6, 10]. The algebraic (syntactic) framework was nicely complemented by a logical (semantical) characterization [6–8, 12, 13, 22]. It turned out that properties definable in counting monadic second-order logic (CMSOL) are recognizable. Moreover, the evaluation mapping as a transduction from terms (tree representations) to graphs can also be defined in CMSOL.

All operations on graphs are deterministic in the papers we have read. Now the question arises whether there are suitable nondeterministic operations which are interesting and for which some of the results still hold. It is not hard to imagine that recognizability could rely on nondeterministic automata [19, 30]. We will consider several variants, but in general they are not equivalent to deterministic automata and therefore lack some closure properties. This seems to destroy the connection with logic unless the nondeterminism is "equational", hence can be simulated with deterministic operations. Unlike recognizability, equational definability (and ground term rewriting) are applicable without change.

Nondeterministic automata can be naturally generalized to produce output. Again, if the results do not depend on the term representations of input and output, they induce a transformation between algebras [25, 15, 18, 17]. We mainly use transducers, the algebraic analogue of linear bottom-up tree transducers. They are more general and possess better closure properties than inductive sets of evaluations, which in turn correspond to linear top-down tree transducers. Transducers are again complemented by a logical notion, namely the definable transductions, which describe transformations of graphs without using term representations. We are able to prove that the effect of

every definable transduction (definable in CMSOL) can be achieved by a transducer. This generalizes the main theorems in [13, 6] and carries the correspondence between definability and recognizability to transformations.

The article is organized as follows. Below we will fix some notations and recall some facts of universal algebra. The algebraic properties of nondeterministic operations are considered in Sections 3 and 4. We compare different formulations of recognizability which are equivalent in the deterministic case. Equational sets are only touched upon shortly. The next section introduces transducers and lists their closure properties. Section 6 introduces finite relational structures together with operations on them. Counting monadic second-order logic is used to express their properties. The heart is Theorem 6.20, which establishes the close connection between both descriptions. The last section tries to give some reasons why monadic second-order logic does not seem to allow unbounded nondeterminism.

2. Preliminaries

After introducing some notation we turn to universal algebra. We define there what we consider as a nondeterministic operation.

2.1. Some notation

We denote by $\mathbb{N}$ the set of nonnegative integers and by $\mathbb{N}_+$ the set of positive ones. We set $[n] = \{1, \ldots, n\}$ for $n \geqslant 0$ (with $[0] = \emptyset$). All numbers in this article are integers, and therefore $n \in \mathbb{N}$ and $n \geqslant 0$ are equivalent statements. The powerset of a set A is written $\mathfrak{P}(A)$. $|A|$ is the cardinality of a set A.

The set of finite sequences of elements from a set A is called A^*. We write sequences as $a_1 \ldots a_n$, $a_i \in A$, $n \geqslant 0$. ε is the empty sequence. The length of a sequence $\vec{a} \in A^*$ is $|\vec{a}|$. If $f: A \to B$ is a mapping, we write $f^*: A^* \to B^*$ for its unique extension to A^*. For $C \subseteq A$, $f \restriction C$ denotes the restriction of f to C.

For a relation R, we use the notations $(a_1, \ldots, a_n) \in R$ and $R(a_1, \ldots, a_n)$ interchangeably. If R is a binary relation, we write $a R b$ instead of $(a, b) \in R$. If R, S are binary relations (or mappings) then $S \circ R$ denotes their composition (first R, then S). R^+ is the transitive closure of R. If R is an equivalence relation on A, $[a]_R$ is the equivalence class of $a \in A$ modulo R and $B/R = \{[b]_R | b \in B\}$ for $B \subseteq A$. Accordingly, $[\,]_R: A \to A/R$ is the canonic surjection. These notations are extended to families of sets and equivalence relations without notice.

We write $S \models \psi$ if S is a structure that satisfies the logical formula ψ. More generally, if ψ contains n free variables $x_1, \ldots, x_n$ in some specified order and $d_1, \ldots, d_n$ are appropriate values for them in S then $S \models \psi(d_1, \ldots, d_n)$ means that S satisfies ψ under the assignment $x_i := d_i, i \in [n]$. We write $\psi[x_1 := t_1, \ldots, x_n := t_n]$ for the simultaneous substitution of terms $t_1, \ldots, t_n$ for variables $x_1, \ldots, x_n$ in ψ. Likewise, $t[c := t_c \mid c \in C]$ is the simultaneous substitution of terms t_c for constants $c \in C$ in t.

2.2. Universal algebra

This subsection recalls some facts about relational magmas (algebras). We formulate definitions and results in terms of category theory. This is not necessary, of course, but shortens the exposition. It may also help to avoid confusion because products (used for product automata in Section 3.1) are not cartesian products.

Definition 2.1 (*continuous, relational, total, functional*). A mapping $f: \mathfrak{P}(M_1) \times \ldots \times \mathfrak{P}(M_n) \to \mathfrak{P}(M)$ is

- *continuous* if $f(\ldots, \bigcup_{i\in\mathbb{N}} X_{ki}, \ldots) = \bigcup_{i\in\mathbb{N}} f(\ldots, X_{ki}, \ldots)$ for every chain $X_{k0} \subseteq \ldots \subseteq X_{ki} \subseteq X_{ki+1} \subseteq \ldots \subseteq M_k$.
- *relational* if it satisfies one of the following three equivalent conditions:
  - $f(\ldots, \bigcup_{i\in I} X_{ki}, \ldots) = \bigcup_{i\in I} f(\ldots, X_{ki}, \ldots)$ for every set I and $X_{ki} \subseteq M_k$. In particular, $f(\ldots, \emptyset, \ldots) = \emptyset$.
  - $f(X_1, \ldots, X_n) = \{m \in M \mid (\exists m_1 \in X_1) \ldots (\exists m_n \in X_n)(m_1, \ldots, m_n, m) \in \tilde{f}\}$ is derived from $\tilde{f} \subseteq M_1 \times \ldots \times M_n \times M$.
  - $f(X_1, \ldots, X_n) = \bigcup_{(m_1,\ldots,m_n)\in X_1\times\ldots\times X_n} \bar{f}(m_1, \ldots, m_n)$ is derived from $\bar{f}: M_1 \times \ldots \times M_n \to \mathfrak{P}(M)$.
- *total* if $f(X_1, \ldots, X_n) = \emptyset$ implies $X_k = \emptyset$ for some $k \in [n]$.
- *functional* if $f(X_1, \ldots, X_n) = \{\bar{f}(m_1, \ldots, m_n) \mid m_k \in X_k, k \in [n]\}$ is derived from $\bar{f}: M_1 \times \ldots \times M_n \to M$.

Functional mappings are relational and total. Compositions of continuous (functional) mappings are continuous (functional). This does not hold for relational or total mappings in general.

Definition 2.2 (*signature*). A signature consists of a set $\mathcal{S}$ of sorts, a set $\mathcal{F}$ of operations, and two mappings $\alpha: \mathcal{F} \to \mathcal{S}^*$ (arguments) and $\rho: \mathcal{F} \to \mathcal{S}$ (result) which together give the type $\alpha(f)\rho(f)$ of $f \in \mathcal{F}$.

We keep the signature fixed in the sequel.

The following definition is *not* standard. It mixes what is normally called a magma with the powerset magma.

Definition 2.3 (*magma, homomorphism*). A magma M consists of a set M_s for every $s \in \mathcal{S}$ and a relational mapping $f_M: \mathfrak{P}(M_{s_1}) \times \ldots \times \mathfrak{P}(M_{s_n}) \to \mathfrak{P}(M_s)$ for every $f \in \mathcal{F}$ of type $s_1 \ldots s_n s$. One can assume that the M_s are pairwise disjoint.

A *relational homomorphism* $\varphi: M \to M'$ consists of a relational mapping $\varphi_s: \mathfrak{P}(M_s) \to \mathfrak{P}(M'_s)$ for every $s \in \mathcal{S}$ such that

$$\varphi_s(f_M(X_1, \ldots, X_n)) = f_{M'}(\varphi_{s_1}(X_1), \ldots, \varphi_{s_n}(X_n))$$

for every $f \in \mathcal{F}$ of type $s_1 \ldots s_n s$ and for all sets $X_i \subseteq M_{s_i}$, $i \in [n]$.

A (*functional*) *homomorphism* $\varphi: M \to M'$ is a relational homomorphism such that each $\varphi_s: \mathfrak{P}(M_s) \to \mathfrak{P}(M'_s)$, $s \in \mathscr{S}$, is functional. Equivalently,

$$(\bar{\varphi}_{s_1}(m_1), \ldots, \bar{\varphi}_{s_n}(m_n), \bar{\varphi}_s(m)) \in \tilde{f}_{M'} \iff (m_1, \ldots, m_n, m) \in \tilde{f}_M$$

for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all $m_i \in M_{s_i}$, $i \in [n]$, $m \in M_s$.

What is usually called an operation is just a functional mapping. Besides that, for example, set intersection is an operation in our sense.

Magmas and (functional or relational) homomorphisms form a category. Such a category is called *functional* or *relational* according to the kind of homomorphisms. Magmas are called total (functional) if all operations are total (functional). A relational category is *total* if magmas and (relational) homomorphisms are total.

Lemma 2.4 (products). *A relational category has all products.*

Notation: The coproduct S of a family S_i, $i \in I$, in the category of sets consists of the disjoint union $\biguplus_{i \in I} S_i$ and inclusions $S_i \to S$, $s \mapsto i.s$.

Proof. The product $M = \prod_{i \in I} M_i$ of a family M_i, $i \in I$, consists of

$$M_s = \biguplus_{i \in I} M_{is}$$

for every $s \in \mathscr{S}$ and

$$f_M: (X_1, \ldots, X_n) \mapsto \biguplus_{i \in I} f_{M_i}(\pi_{is_1}(X_1), \ldots, \pi_{is_n}(X_n))$$

for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$, together with projections π_i, $i \in I$, where

$$\pi_{is}: X \mapsto \{m \in M_{is} \mid i.m \in X\}$$

for every $s \in \mathscr{S}$. A family of relational homomorphisms $\varphi_i: L \to M_i, i \in I$, induces the relational homomorphism $\varphi: L \to M$ such that

$$\varphi_s: X \mapsto \biguplus_{i \in I} \varphi_{is}(X)$$

for every $s \in \mathscr{S}$. □

We write $M_1 \times M_2$ for $\prod_{i \in \{1,2\}} M_i$ as usual, although it is not the cartesian product.

Let us note that a total category still has a terminal object M, which is different, however: $M_s = \{s\}$ for every $s \in \mathscr{S}$, and the operations are fully determined.

Lemma 2.5 (term magma). *A relational category has an initial object, called the* term magma $\mathscr{T}(\mathscr{F})$.

It also has all coproducts, but we will not need them.

Proof. $\mathscr{T}(\mathscr{F})$ consists of

$$\mathscr{T}_s = \{ft_1 \dots t_n \mid f \in \mathscr{F}, \rho(f) = s, \alpha(f) = s_1 \dots s_n, t_k \in \mathscr{T}_{s_k}, k \in [n]\}$$

(defined simultaneously) for every $s \in \mathscr{S}$, and

$$f_{\mathscr{T}}: (X_1, \dots, X_n) \mapsto \{ft_1 \dots t_n | t_k \in X_k, k \in [n]\}$$

for every $f \in \mathscr{F}$ of type $s_1 \dots s_n s$. The initial relational homomorphism $\mathsf{val}: \mathscr{T}(\mathscr{F}) \to M$ evaluates terms in M. □

Definition 2.6 (*generate*). The initial relational homomorphism $\mathsf{val}: \mathscr{T}(\mathscr{F}) \to M$ generates M if $M_s = \bigcup_{t \in \mathscr{T}_s(\mathscr{F})} \mathsf{val}(\{t\})$ for every $s \in \mathscr{S}$.

Now we extend the notation for the term magma.

Definition 2.7 (*term, derived operation*). Let $x_1 \dots x_k$, $k \geqslant 0$, be a sequence (without repetitions) of *variables* together with a mapping $\rho: \{x_1, \dots, x_k\} \to \mathscr{S}$. The set of *terms* of sort $s \in \mathscr{S}$, $\mathscr{T}_s(\mathscr{F}, x_1, \dots, x_k)$, is defined (simultaneously for all $s \in \mathscr{S}$) as the smallest set containing

$$\{x \in \{x_1, \dots, x_k\} | \rho(x) = s\}$$

and

$$\{ft_1 \dots t_n | f \in \mathscr{F}, \rho(f) = s, \alpha(f) = s_1 \dots s_n, t_i \in \mathscr{T}_{s_i}(\mathscr{F}, x_1, \dots, x_k), i \in [n]\}$$

as subsets.

A term is *linear* if it contains every variable at most once.

A term t of sort $s \in \mathscr{S}$ over $x_1 \dots x_k$ induces a *derived operation* $M(t): \mathfrak{P}(M_{\rho(x_1)}) \times \dots \times \mathfrak{P}(M_{\rho(x_k)}) \to \mathfrak{P}(M_s)$ on every magma M according to the following inductive definition: $M(x_i): (X_1, \dots, X_k) \mapsto X_i$ is the ith projection and $M(ft_1 \dots t_n) = f_M \circ (M(t_1), \dots, M(t_n))$ the composition.

Immediate consequences are that

$$M(t[x_1 := t_1, \dots, x_n := t_n]) = M(t) \circ (M(t_1), \dots, M(t_n))$$

and $\varphi_s \circ M(t) = M'(t) \circ (\varphi_{s_1}, \dots, \varphi_{s_n})$ for a relational homomorphism $\varphi: M \to M'$. Linear terms induce relational operations.

Definition 2.8 (*context*). A *context* c of type $s_1 s_2$ in a magma M is a (relational) mapping

$$c: \mathfrak{P}(M_{s_1}) \to \mathfrak{P}(M_{s_2}),\ X \mapsto M(t)(X, X_2, \dots, X_n)$$

such that $t \in \mathscr{T}_{s_2}(\mathscr{F}, x_1, \dots, x_n)$ is linear, $\rho(x_1) = s_1$, and $X_i \subseteq M_{\rho(x_i)}$, $i \in [n] \setminus [1]$. We write $\varphi(c)$ for the mapping

$$\mathfrak{P}(M'_{s_1}) \to \mathfrak{P}(M'_{s_2}),\ X \mapsto M'(t)(X, \varphi_{s_2}(X_2), \dots, \varphi_{s_n}(X_n))$$

where $\varphi: M \to M'$ is a relational homomorphism.

3. Recognizable sets

We will introduce four notions of recognizability and compare them. The first two, $\exists$-recognizability and $\forall$-recognizability by nondeterministic automata, lack closure under complement. Both notions are dual to each other and both have an equivalent formulation as a set of predicates and as syntactic equivalences. Their main purpose is to state this equivalence in full generality. A similar statement can be made for their common restriction, which can be formulated with congruence relations. Recognizability by deterministic automata, the fourth concept, is already very similar to the standard notion. The reader may want to confer [10] as a standard reference.

3.1. Nondeterministic recognizability

Definition 3.1 (*locally finite magma, automaton, recognizable set*). A magma M is *locally finite* if M_s is finite for every $s \in \mathscr{S}$.

A (*nondeterministic*) *automaton* on a magma M is a triple (σ, A, F), where $\sigma : M \to A$ is a relational homomorphism, A is a locally finite magma, and $F \subseteq A_s$ for some $s \in \mathscr{S}$.

A set $R \subseteq M_s$ is $\exists$-*recognizable* if $R = \{m \in M_s \mid \sigma_s(\{m\}) \cap F \neq \emptyset\}$. A set $R \subseteq M_s$ is $\forall$-*recognizable* if $R = \{m \in M_s \mid \sigma_s(\{m\}) \subseteq F\}$.

Definition 3.2 (*set of predicates*). A set of predicates $\mathscr{P}$ for a magma M is given by a mapping $\alpha : \mathscr{P} \to \mathscr{S}$ ($\alpha(p)$ is called the *sort* of p) and an interpretation $\hat{p} \subseteq M_{\alpha(p)}$ for every $p \in \mathscr{P}$.

A set of predicates $\mathscr{P}$ is *locally finite* if $\mathscr{P}_s = \{p \in \mathscr{P} \mid \alpha(p) = s\}$ is finite for every $s \in \mathscr{S}$.

A set of predicates $\mathscr{P}$ is $\exists$-$\mathscr{F}$-*inductive* if, for every $p \in \mathscr{P}_s$ and every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$, there exist $k \in \mathbb{N}$ and $p_{i1}, \ldots, p_{ik} \in \mathscr{P}_{s_i}$, $i \in [n]$, such that for all $X_i \subseteq M_{s_i}$, $i \in [n]$: $f_M(X_1, \ldots, X_n) \cap \hat{p} \neq \emptyset$ if and only if

$$\bigvee_{i \in [k]} \bigwedge_{j \in [n]} X_j \cap \hat{p}_{ji} \neq \emptyset .$$

Similarly, a set of predicates $\mathscr{P}$ is $\forall$-$\mathscr{F}$-*inductive* if for all $X_i \subseteq M_{s_i}$, $i \in [n]$: $f_M(X_1, \ldots, X_n) \subseteq \hat{p}$ if and only if

$$\bigwedge_{i \in [k]} \bigvee_{j \in [n]} X_j \subseteq \hat{p}_{ji}$$

or, equivalently, $\mathscr{P}$ under the complementary interpretation ($p \in \mathscr{P}$ is interpreted by $M_{\alpha(p)} \setminus \hat{p}$) is $\exists$-$\mathscr{F}$-*inductive*.

The sequence $p_{11} \ldots p_{1k} \ldots p_{n1} \ldots p_{nk}$ is called a *decomposition* of p relative to f in both cases.

Definition 3.3 (*syntactic equivalence*). Let M be a magma and $R \subseteq M_r$. The *syntactic* $\exists$-*equivalence* with respect to R is the family of relations $\sim_s$ on $\mathfrak{P}(M_s)$, $s \in \mathscr{S}$, such

that $X \sim_s X'$ if, for all contexts c of type sr, $c(X) \cap R \neq \emptyset \iff c(X') \cap R \neq \emptyset$. The *syntactic* $\forall$*-equivalence* with respect to R is the family of relations $\sim_s$ on $\mathfrak{P}(M_s)$, $s \in \mathscr{S}$, such that $X \sim_s X'$ if, for all contexts c of type sr, $c(X) \subseteq R \iff c(X') \subseteq R$.

A family of equivalence relations $\sim_s$ on $\mathfrak{P}(M_s)$, $s \in \mathscr{S}$, is *locally finite* if $\mathfrak{P}(M_s)/\sim_s$ is finite for every $s \in \mathscr{S}$.

Proposition 3.4 (equivalence 1). *Let Q range over $\{\exists, \forall\}$. The following conditions for a set R are equivalent:*
(1) *R is Q-recognizable by a (nondeterministic) automaton (σ, A, F) on M.*
(2) *$R = \hat{p}$, $p \in \mathscr{P}$, where $\mathscr{P}$ is a locally finite, Q-$\mathscr{F}$-inductive set of predicates for M.*
(3) *The syntactic Q-equivalence $\sim$ with respect to R is locally finite.*

(1) $\Rightarrow$ (2) and (2) $\Rightarrow$ (3) are as in [10]. (2) $\Rightarrow$ (1) is simple and uses nondeterminism. We only consider the case $Q = \exists$.

Proof. (1) implies (2): Let $R \subseteq M_r$ and define

$$\mathscr{P} = \{p\} \uplus \biguplus_{s \in \mathscr{S}} A_s ,$$

$\alpha(p) = r$, $\alpha(a) = s$ for $a \in A_s$,

$$\hat{p} = \{m \in M_r \mid \sigma_r(\{m\}) \cap F \neq \emptyset\} ,$$

$$\hat{a} = \{m \in M_{\alpha(a)} \mid a \in \sigma_{\alpha(a)}(\{m\})\} .$$

We have

$$\begin{aligned} & f_M(X_1, \ldots, X_n) \cap \hat{a} \neq \emptyset \\ & \quad \iff a \in \sigma_s(f_M(X_1, \ldots, X_n)) = f_A(\sigma_{s_1}(X_1), \ldots, \sigma_{s_n}(X_n)) \\ & \quad \iff \bigvee_{\substack{a_1 \in A_{s_1}, \ldots, a_n \in A_{s_n} \\ a \in f_A(\{a_1\}, \ldots, \{a_n\})}} X_1 \cap \hat{a}_1 \neq \emptyset \wedge \ldots \wedge X_n \cap \hat{a}_n \neq \emptyset \end{aligned}$$

for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$, and similarly for $\hat{p}$.

(2) implies (1): Let

$$A_s = \mathscr{P}_s ,$$

$$f_A : (X_1, \ldots, X_n) \mapsto \{a \subset \mathscr{P} \mid \bigvee_{i \in [k]} \bigwedge_{j \in [n]} a_{ji} \subset X_i\} ,$$

where $a_{11} \ldots a_{1k} \ldots a_{n1} \ldots a_{nk}$ is a decomposition of a relative to f. The mapping f_A is relational. It is easy to check that

$$\sigma_s : X \mapsto \{a \in \mathscr{P}_s \mid X \cap \hat{a} \neq \emptyset\}$$

is a relational homomorphism from M to A. So we can set $F = \{p\}$.

(2) implies (3): Define a family of equivalence relations $\approx_s$ on $\mathfrak{P}(M_s)$, $s \in \mathscr{S}$:

$$X \approx_s X'$$

means that, for all $a \in \mathscr{P}_s$,

$$X \cap \hat{a} \neq \emptyset \iff X' \cap \hat{a} \neq \emptyset .$$

This family is locally finite and refines the syntactic $\exists$-equivalence.

(3) implies (2): Let $R \subseteq M_r$ and define an equivalence relation $\approx_s$ on the set of contexts of type sr, $s \in \mathscr{S}$:

$$c_1 \approx_s c_2$$

means that, for all $X \subseteq M_s$,

$$c_1(X) \cap R \neq \emptyset \iff c_2(X) \cap R \neq \emptyset .$$

(Actually, we have to test this only for the finitely many classes $[X]_{\sim_s}$ in $\mathfrak{P}(M_s)/\sim_s$.) Let $\mathscr{R}_s$ be the finite set of all contexts of type sr modulo $\approx_s$ and define

$$\hat{a} = \{m \in M_s | c(\{m\}) \cap R \neq \emptyset, c \in a\} .$$

We have

$$f_M(X_1,\ldots,X_n) \cap \hat{a} \neq \emptyset \iff c(f_M(X_1,\ldots,X_n)) \cap R \neq \emptyset, \quad c \in a$$

for each $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and $a \in \mathscr{P}_s$. (Clearly, the choice of $c \in a$ does not matter.) The outcome depends only on $[f_M(X_1,\ldots,X_n)]_{\sim_s}$, hence allows only finitely many classes $[X_i]_{\sim_{s_i}}$, $i \in [n]$. Consider the jth combination of representatives $X_{1j} \subseteq M_{s_1}$, …, $X_{nj} \subseteq M_{s_n}$ and let a_{ij}, $i \in [n]$, be the class of context $X_{ij} \mapsto c(f_M(X_{1j},\ldots,X_{nj}))$ (of type $s_i r$) modulo $\approx_{s_i}$, $c \in a$. Then the condition above is equivalent to

$$\bigvee_j \bigwedge_{i \in [n]} X_i \cap \hat{a}_{ij} \neq \emptyset .$$

This proves $\exists$-$\mathscr{F}$-inductivity. Finally, $R = \hat{p}$, where p is the equivalence class of the identity (defined by the term $x \in \mathscr{T}_r(\mathscr{F}, x)$). □

$\emptyset$ is $\exists$-recognizable, and M_s is $\forall$-recognizable (take the terminal object for A, the induced relational homomorphism for σ, and $F = \emptyset$), but not necessarily vice versa (Example 4.3 below). The complement of an $\exists$-recognizable set is $\forall$-recognizable and vice versa: Just take the complement of F. We therefore restrict our attention to $\exists$-recognizable sets.

If (σ_1, A_1, F_1) $\exists$-recognizes $R_1 \subseteq M_s$ and (σ_2, A_2, F_2) $\exists$-recognizes $R_2 \subseteq M_s$ then $(\sigma, A_1 \times A_2, F_1 \uplus F_2)$ $\exists$-recognizes $R_1 \cup R_2$, where σ is the induced relational homomorphism. There is no such construction to prove closure under intersection.

If $\varphi : M \to M'$ is a relational homomorphism and (σ, A, F) on M' $\exists$-recognizes $R' \subseteq M'_s$ then $(\sigma \circ \varphi, A, F)$ on M $\exists$-recognizes $\{m \in M_s \mid \varphi_s(\{m\}) \cap R' \neq \emptyset\}$.

If (σ, A, F) $\exists$-recognizes $R \subseteq M_r$ and c is a context of type sr then

$$(\sigma, A, \{a \in A_s \mid \sigma(c)(\{a\}) \cap F \neq \emptyset\})$$

$\exists$-recognizes $\{m \in M_s \mid c(\{m\}) \cap R \neq \emptyset\}$.

Definition 3.5 (*congruence*). A *congruence relation* $\approx$ on a magma M consists of an equivalence relation $\approx_s$ on M_s for every $s \in \mathscr{S}$ such that $X_i/\approx_{s_i} = X_i'/\approx_{s_i}$ implies $f_M(X_1, \ldots, X_n)/\approx_s = f_M(X_1', \ldots, X_n')/\approx_s$ for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all sets $X_i, X_i' \subseteq M_{s_i}$, $i \in [n]$.

A congruence relation $\approx$ is *locally finite* if $M/\approx$ is locally finite.

A set $R \subseteq M_s$ is $\approx$-*saturated* if $m \in R$ and $m \approx_s m'$ imply $m' \in R$ for all $m, m' \in M_s$. Equivalent conditions are: $X/\approx_s \cap R/\approx_s \neq \emptyset$ implies $X \cap R \neq \emptyset$ for all $X \subseteq M_s$, and $X/\approx_s \subseteq R/\approx_s$ implies $X \subseteq R$ for all $X \subseteq M_s$.

Remark 3.6. A syntactic $\exists$-equivalence or $\forall$-equivalence $\sim$ consists of congruence relations $\sim_s$, $s \in \mathscr{S}$, on $\mathfrak{P}(M_s)$ in the usual sense – they are compatible with the operations of M. But $\sim$ is not necessarily a congruence relation according to the preceding definition, because there may be equivalence classes with respect to $\sim_s$ that do not contain singleton sets.

Remark 3.7. Congruence relations are some kind of *bisimulations*. Consider, for the sake of simplicity, a magma M over a signature with one sort s and such that all operations are unary. Then an operation f_M, $f \in \mathscr{F}$, is simply a binary relation on M_s. (Such a magma M is often called a *labelled transition system* or *Kripke structure*, the elements of M_s are the *states* or *possible worlds* and the $\bar{f}_M$ are the *transition* or *accessibility* relations.) Two elements $m_1, m_2 \in M_s$ are bisimilar if, for all $f \in \mathscr{F}$, whenever $m_1 \; \bar{f}_M \; m_1'$ for some $m_1' \in M_s$ then $m_2 \; \bar{f}_M \; m_2'$ for some $m_2' \in M_s$ such that m_1', m_2' are bisimilar, and vice versa, whenever $m_2 \; \bar{f}_M \; m_2'$ for some $m_2' \in M_s$ then $m_1 \; \bar{f}_M \; m_1'$ for some $m_1' \in M_s$ such that m_1', m_2' are bisimilar. Obviously, if $\approx$ is a congruence relation on M and $m_1 \approx_s m_2$ then they are bisimilar.

With respect to a (nondeterministic) automaton (σ, A, F), where $\sigma : M \to A$ is a (functional) homomorphism, a set $R \subseteq M_s$ is $\exists$-recognizable if and only if it is $\forall$-recognizable if and only if $R = \{m \in M_s \mid \bar{\sigma}_s(m) \in F\}$.

Lemma 3.8 (equivalence 2). *The following conditions for a set R are equivalent:*

(1) *R is recognizable by a (nondeterministic) automaton (σ, A, F) on M, where σ is a (functional) homomorphism.*

(2) *R is $\approx$-saturated, where $\approx$ is a locally finite congruence relation on M.*

The proof is as in [10].

Proof. (1) implies (2): R is $\approx$-saturated, where $m \approx_s m'$ means $\bar{\sigma}_s(m) = \bar{\sigma}_s(m')$.
(2) implies (1): The (nondeterminstic) automaton $([\,]_\approx, M/\approx, R/\approx_s)$ recognizes R. □

It follows that the class of such sets is closed under complement and inverse homomorphisms. $\emptyset$ and M_s are, however, not necessarily recognizable.

Examples for recognizable sets are given below.

3.2. Deterministic recognizability

Definition 3.9 (*deterministic automaton*). An automaton (σ, A, F) on a (not necessarily functional) magma M is *deterministic* if $\sigma: M \to A$ is a (functional) homomorphism and A is a functional magma.

Deterministic recognizability maintains all the closure properties stated above, namely closure under complement, union, intersection, inverse homomorphisms and inverse contexts. The reason is that closure with respect to union and intersection can be proved with the usual (cartesian!) product automaton.

With deterministic recognizability the syntactic equivalence recovers its classical meaning.

Proposition 3.10 (equivalence 3). *Let $\sim$ be the syntactic $\exists$-equivalence ($\forall$-equivalence) with respect to a set $R \subseteq M_r$ and define a relation $\approx$ on M by $m \approx_s m' \iff \{m\} \sim_s \{m'\}$ for all $m, m' \in M_s$ and for every $s \in \mathscr{S}$. The following five conditions for R are equivalent:*

(1) *R is deterministically recognizable.*

(2) (3) *The syntactic $\exists$-equivalence ($\forall$-equivalence) with respect to R is locally finite and satisfies the following condition: $m, m' \in f_M(\{m_1\}, \ldots, \{m_n\})$ implies $m \approx_s m'$ for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all $m_i \in M_{s_i}$, $i \in [n]$.*

(4) (5) *The syntactic $\exists$-equivalence ($\forall$-equivalence) with respect to R is locally finite and $\approx$ is a congruence relation such that $f_M(\{m_1\}, \ldots, \{m_n\})/\approx_s = \{[f_M(\{m_1\}, \ldots, \{m_n\})]_{\sim_s}\}$ for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all $m_i \in M_{s_i}$, $i \in [n]$.*

Proof. (1) implies (2): We only consider the syntactic $\exists$-equivalence, but the same proof works for the syntactic $\forall$-equivalence. Let (σ, A, F) be a deterministic automaton on M. We have

$$\bar{\sigma}_s(m), \bar{\sigma}_s(m') \in \sigma_s(f_M(\{m_1\}, \ldots, \{m_n\})) = f_A(\sigma_{s_1}(\{m_1\}), \ldots, \sigma_{s_n}(\{m_n\}))\,,$$

hence $\bar{\sigma}_s(m) = \bar{f}_A(\bar{\sigma}_{s_1}(m_1), \ldots, \bar{\sigma}_{s_n}(m_n)) = \bar{\sigma}_s(m')$. It follows that $\overline{\sigma(c)}(\bar{\sigma}_s(m)) = \overline{\sigma(c)}(\bar{\sigma}_s(m'))$ for every context c of type sr. Therefore,

$$c(\{m\}) \cap R \neq \emptyset \iff \overline{\sigma(c)}(\bar{\sigma}_s(m)) \in F$$
$$\iff \overline{\sigma(c)}(\bar{\sigma}_s(m')) \in F \iff c(\{m'\}) \cap R \neq \emptyset$$

as required.

(2) implies (4): By assumption,

$$f_M(\{m_1\}, \ldots, \{m_n\})/\approx_s = \{[m]_{\approx_s}\} = \{[\{m\}]_{\sim_s}\} = \{[f_M(\{m_1\}, \ldots, \{m_n\})]_{\sim_s}\}$$

for any $m \in f_M(\{m_1\}, \ldots, \{m_n\})$. It is easy to see that $\approx$ is a congruence relation.

(4) implies (1): The following automaton (σ, A, F) is deterministic and recognizes R: $A_s = M_s/\approx_s$, $\tilde{f}_A: (a_1, \ldots, a_n) \mapsto [f_M(a_1, \ldots, a_n)]_{\sim_s}$ (if $\rho(f) = s$), $\tilde{\sigma}_s: m \to [m]_{\approx_s}$ and $F = R/\approx_r$. □

On a functional magma all notions of recognizability coincide and give the standard notion of recognizability for terms.

Lemma 3.11 (equivalence 4). *If M is a functional magma then the following conditions for $R \subseteq M_s$ are equivalent:*
(1) *R is $\exists$-recognizable.*
(2) *R is $\forall$-recognizable.*
(3) *R satisfies the conditions of the preceding lemma.*
(4) *R is deterministically recognizable.*

We employ the usual subset construction. In essence, this lemma was already proved in [30].

Proof. Obviously, (4) implies (3), (3) implies (1) and (2). For the converse consider a (nondeterministic) automaton (σ, A, F). We construct an equivalent deterministic automaton (σ', A', F') as follows:
- $\tilde{\sigma}'_s(m) = \sigma_s(\{m\})$ for every $m \in M_s$ and $s \in \mathcal{S}$,
- $A'_s = \mathfrak{P}(A_s)$ for every $s \in \mathcal{S}$,
- $\tilde{f}_{A'} = f_A$ for every $f \in \mathcal{F}$.
- F' depends on the starting point: For 1, $F' = \{X \subseteq M_s | X \cap F \neq \emptyset\}$. For 2, $F' = \{X \subseteq M_s | X \subseteq F\}$.

Obviously, (σ', A', F') recognizes R. □

Example 3.12 (*recognizable sets*). Let $\mathcal{S} = \{s\}$, $\mathcal{F} = \{f\}$ and f have type ss.
(1) Consider the magma M with $M_s = \mathbb{Z}$ and $f_M: \{n\} \mapsto \{n-1, n+1\}$. $2\mathbb{Z}$ is deterministically recognizable. A deterministic automaton (σ, A, F) is given by $A_s = \{0, 1\}$, $\tilde{f}_A$ as Boolean negation, $\tilde{\sigma}_s: n \mapsto n \bmod 2$, and $F = \{0\}$. Actually, the amount of nondeterminism in f_M does not matter as long as all elements in the image have some property in common.
(2) Consider the magma M with $M_s = \mathbb{N}$ and $f_M: \{n\} \mapsto \{n, n+1\}$. The sets $R_k = \{n \in N | n \geqslant k\}$, $k \geqslant 0$, are all $\exists$-recognizable. A (nondeterministic) automaton (σ, A, F) with a homomorphism σ is given by $A_s = \{0, \ldots, k\}$,

$$f_A: \{n\} \mapsto \begin{cases} \{k\} & \text{if } n = k \\ \{n, n+1\} & \text{otherwise} \end{cases}, \qquad \tilde{\sigma}_s: n \mapsto \begin{cases} n & \text{if } n \leqslant k \\ k & \text{otherwise} \end{cases}$$

and $F = \{k\}$. R_k is also $\forall$-recognizable (by Lemma 3.8) but not deterministically recognizable (by Proposition 3.10) – the elements of $\{0, \ldots, k\}$ are distinguishable by their distance to k.

(3) Consider the same M as before. The sets $\{k\}$, $k \geqslant 0$, are all $\exists$-recognizable. A (nondeterministic) automaton (σ, A, F) is given by the same A, F as before and

$$\sigma_s\colon \{n\} \mapsto \begin{cases} \{n\} & \text{if } n \leqslant k \\ \emptyset & \text{otherwise} \end{cases}.$$

The sets $\{k\}$, $k \geqslant 0$, are also $\forall$-recognizable, because the $\forall$-equivalence is locally finite. A (nondeterministic) automaton (σ, A, F) is given by $A_s = \{0,1\}$, $\bar{f}_A\colon n \mapsto 0$,

$$\sigma_s\colon \{n\} \mapsto \begin{cases} \{1\} & \text{if } n = k \\ \{0,1\} & \text{otherwise} \end{cases}$$

and $F = \{1\}$.

(4) Consider the magma M with $M_s = \mathbb{N}$ and $f_M\colon \{n\} \mapsto \{0,\ldots,n-1\}$. None of the sets $R_k = \{n \in N \mid n \leqslant k\}$, $k \geqslant 0$, is $\exists$-recognizable ($\forall$-recognizable), because the syntactic $\exists$-equivalence ($\forall$-equivalence) is not locally finite.

All kinds of recognizable sets are inherited by submagmas and (the same magmas with respect to) subsignatures. In general, only deterministically recognizable sets are inherited by derived magmas, that is, magmas with derived operations.

As stated in the introduction, algebraic automata can be considered as tree automata running on term representations. The concepts presented here, however, work without this description. This is contrary to intuition: If a magma contains elements which are not values of terms, their acceptance does not depend on any "run". A triple (σ, A, F) could then hardly be called an automaton. For similar reasons one might be tempted to allow only total magmas. This is quite natural, but one should not be too restrictive. The rest of this subsection will show that automata must be allowed to discriminate between different values of the same term. Otherwise, nondeterminism is essentially prevented.

We consider a total magma M which is generated by the initial relational homomorphism $\mathsf{val}\colon \mathcal{T}(\mathcal{F}) \to M$. Let $\sim$ be the relation on M such that $m \sim_s m'$, $m, m' \in M_s$, means that there exists $t \in \mathcal{T}_s(\mathcal{F})$ with $m, m' \in \mathsf{val}(\{t\})$. We show that its transitive closure $\sim^+$ (separately for each sort) is a congruence relation. Assume that there exist $t_0,\ldots,t_k \in \mathcal{T}_s(\mathcal{F})$, $k \geqslant 0$, such that $m \in \mathsf{val}(\{t_0\})$, $\mathsf{val}(\{t_{i-1}\}) \cap \mathsf{val}(\{t_i\}) \neq \emptyset$, $i \in [k]$, and $m' \in \mathsf{val}(\{t_k\})$. We have to show that

$$f_M(\{m_1\},\ldots,\{m\},\ldots,\{m_n\})/{\sim_s^+} = f_M(\{m_1\},\ldots,\{m'\},\ldots,\{m_n\})/{\sim_s^+}$$

for all $m_i \in M_{s_i}$, $i \in [n]$ (without one position). Find $t_i' \in \mathcal{T}_{s_i}(\mathcal{F})$ such that $m_i \in \mathsf{val}(\{t_i'\})$, $i \in [n]$ (without one position). It follows that

$$\begin{aligned} &f_M(\{m_1\},\ldots,\{m\},\ldots,\{m_n\}) \\ &\quad \subseteq f_M(\mathsf{val}(\{t_1'\}),\ldots,\mathsf{val}(\{t_0\}),\ldots,\mathsf{val}(\{t_n'\})) = \mathsf{val}(\{f t_1' \ldots t_0 \ldots t_n'\}), \\ &\mathsf{val}(\{f t_1' \ldots t_{i-1} \ldots t_n'\}) \cap \mathsf{val}(\{f t_1' \ldots t_i \ldots t_n'\}) \\ &\quad \supseteq f_M(\mathsf{val}(\{t_1'\}),\ldots,\mathsf{val}(\{t_{i-1}\}) \cap \mathsf{val}(\{t_i\}),\ldots,\mathsf{val}(\{t_n'\})) \neq \emptyset, \end{aligned}$$

$i \in [k]$, and

$$f_M(\{m_1\},\ldots,\{m'\},\ldots,\{m_n\}) \subseteq f_M(\mathsf{val}(\{t_1'\}),\ldots,\mathsf{val}(\{t_k\}),\ldots,\mathsf{val}(\{t_n'\})) = \mathsf{val}(\{f t_1' \ldots t_k \ldots t_n'\}) .$$

Obviously, $M/\sim^+$ is a functional magma.

Lemma 3.13 (equivalence 5). *Let M be a total magma which is generated by the initial relational homomorphism* $\mathsf{val}: \mathscr{T}(\mathscr{F}) \to M$. *Then the following conditions for $R \subseteq M_s$ are equivalent:*
(1) $\{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \cap R \neq \emptyset\} \subseteq \{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\}$.
(2) *R is saturated with respect to the congruence relation $\sim^+$.*

Proof. (1) implies (2): Assume that $m \in R$ and $m \sim_s m'$. There exists $t \in \mathscr{T}_s(\mathscr{F})$ such that $m, m' \in \mathsf{val}(\{t\})$. $\mathsf{val}(\{t\}) \subseteq R$ because $m \in \mathsf{val}(\{t\}) \cap R$. Therefore, $m' \in R$.
(2) implies (1): If $m \in \mathsf{val}(\{t\}) \cap R$ then $m' \in \mathsf{val}(\{t\})$ implies $m \sim_s m'$ and $m' \in R$. □

Proposition 3.14 (equivalence 6). *Let the initial relational homomorphism* val: $\mathscr{T}(\mathscr{F}) \to M$ *be total and generate M. Let M and $R \subseteq M_s$ satisfy the conditions of the preceding lemma. Then the following conditions are equivalent:*
(1) $\{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\}$ *is (deterministically) recognizable.*
(2) $R/\sim_s^+$ *is (deterministically) recognizable.*

We are in the situation $\mathscr{T}(\mathscr{F}) \xrightarrow{\mathsf{val}} M \xrightarrow{[\,]_{\sim^+}} M/\sim^+$, where $[\,]_{\sim^+} \circ \mathsf{val}$ is a homomorphism by the totality of val and the definition of $\sim$.

Proof. (1) implies (2): Assume that the deterministic automaton (σ, A, F) on $\mathscr{T}(\mathscr{F})$ recognizes $\{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\}$. We define a locally finite congruence relation $\approx^+$ on M:

$$m \approx_s m'$$

means that there exist $t, t' \in \mathscr{T}_s(\mathscr{F})$ such that

$$m \in \mathsf{val}(\{t\}),\ \bar{\sigma}_s(t) = \bar{\sigma}_s(t'),\ m' \in \mathsf{val}(\{t'\}) .$$

R is $\approx^+$-saturated: Assume $m \approx_s m'$ and take t, t' as above. Then, $m \in R$ implies

$$t \in \{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \cap R \neq \emptyset\} \subseteq \{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\} .$$

Therefore, $\bar{\sigma}_s(t') = \bar{\sigma}_s(t) \in F$ and further $t' \in \{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\}$. Finally, $m' \in \mathsf{val}(\{t'\}) \subseteq R$. Since $\sim^+$ refines $\approx^+$, $[\,]_{\approx^+}$ can be factored over $[\,]_{\sim^+}$.

(2) implies (1): We show that the deterministic automaton $(\sigma \circ [\,]_{\sim^+} \circ \mathsf{val}, A, F)$ on $\mathscr{T}(\mathscr{F})$ recognizes $\{t \in \mathscr{T}_s(\mathscr{F}) \mid \mathsf{val}(\{t\}) \subseteq R\}$ if the deterministic automaton (σ, A, F) on $M/\sim^+$ recognizes $R/\sim_s^+$. If $\mathsf{val}(\{t\}) \subseteq R$, $t \in \mathscr{T}_s(\mathscr{F})$, then

$$\sigma_s(\mathsf{val}(\{t\})/\sim_s^+) \subseteq \sigma_s(R/\sim_s^+) = F .$$

On the other hand, if $\sigma_s(\mathsf{val}(\{t\})/{\sim_s^+}) \subseteq F$ then $\mathsf{val}(\{t\})/{\sim_s^+} \subseteq R/{\sim_s^+}$, and $\mathsf{val}(\{t\}) \subseteq R$ because R is $\sim^+$-saturated. □

Remark 3.15. (1) implies (2) still holds if val is not total. One can say that, intuitively, every set which is recognizable by a tree automaton running on its preimage in the term magma is itself the preimage of a recognizable set in a functional magma.

Example 3.16 (*improper nondeterminism*). Let us review the first item of Example 3.12 and extend $\mathscr{F}$ by a constant c of type s. If we interpret it as $c_M = \{0\}$ then $-i \sim_s i$ for all $i \in \mathbb{N}$: Both are values of the term $f^i c$. $M/{\sim^+}$ is isomorphic to the functional magma M' with $M'_s = \mathbb{N}$ and $f_{M'}$ the successor mapping. If $c_M = 2\mathbb{Z}$ then $M/{\sim^+}$ is even isomorphic to A (extended by $\bar{c}_A = 0$). By Lemmas 3.13 and 3.8, $R = 2\mathbb{Z}$ is $\exists$-recognizable and $\forall$-recognizable.

In both cases, Proposition 3.14 also applies. Not only R is deterministically recognizable in M but also its preimage $\{f^i c \mid i \in 2\mathbb{N}\}$ in $\mathscr{T}_s(\mathscr{F})$ and its canonic image $R/{\sim_s^+}$ in $M/{\sim^+}$ are deterministically recognizable. The elements of M_s can be seen as copies of the "true" values in $M_s/{\sim_s^+}$ of the terms $f^i c$, $i \in \mathbb{N}$.

4. Equational sets

Systems of equations normally lift all operations to the powerset magma. That these lifted operations are continuous is all what is really needed. Now we are working already in such a powerset structure, with continuous operations. Therefore, the theory goes through virtually without change.

Definition 4.1 (*expression, derived operation*). The set of *expressions* of sort $s \in \mathscr{S}$ over $x_1 \ldots x_k$, $\mathscr{E}_s(\mathscr{F}, x_1, \ldots, x_k)$, is the smallest set containing

$$\mathscr{T}_s(\mathscr{F}, x_1, \ldots, x_k)$$

$$\{\Omega_s\}$$

$$\{e_1 +_s e_2 \mid e_1, e_2 \in \mathscr{E}_s(\mathscr{F}, x_1, \ldots, x_k)\}$$

as subsets. (Ω_s and $+_s$ for each $s \in \mathscr{S}$ can be seen as additional quasi-operations of type s and sss, respectively, which have a fixed interpretation. However, $+_s$ is not relational, because it is not *strict*, i.e., it does not preserve the empty set.) These expressions are also called *polynomials* and the terms in them *monomials*.

An expression e of sort $s \in \mathscr{S}$ over $x_1 \ldots x_k$ induces a *derived operation* $M(e)$: $\mathfrak{P}(M_{\rho(x_1)}) \times \ldots \times \mathfrak{P}(M_{\rho(x_k)})) \to \mathfrak{P}(M_s)$ on every magma M according to the following inductive definition extending $M(t)$ for terms $t \in \mathscr{T}_s(\mathscr{F}, x_1, \ldots, x_k)$: $M(\Omega_s)$: $(X_1, \ldots, X_k) \mapsto \emptyset$ is the empty set and

$$M(e_1 +_s e_2)\colon (X_1, \ldots, X_k) \mapsto M(e_1)(X_1, \ldots, X_k) \cup M(e_2)(X_1, \ldots, X_k)$$

is set union.

We note that relational homomorphisms preserve the operations Ω_s and $+_s$, and that $+_s$ is a commutative, associative operation with neutral element Ω_s.

Definition 4.2 (*equational set*). A *system of equations* E is a sequence

$$x_1 = e_1, \quad \ldots, \quad x_n = e_n ,$$

where $e_i \in \mathscr{E}_{\rho(x_i)}(\mathscr{F}, x_1, \ldots, x_n)$, $i \in [n]$.

The *least solution* $\mathscr{L}(E,M)$ of E *in a magma* M is the least fixpoint of the mapping

$$E_M : (X_1, \ldots, X_n) \mapsto (M(e_1)(X_1, \ldots, X_n), \ldots, M(e_n)(X_1, \ldots, X_n)) ,$$

where $X_i \subseteq M_{\rho(x_i)}$, $i \in [n]$. (It exists because the $M(e_i)$ are continuous.) We write $\mathscr{L}(E,M).x_i$ for the ith component of $\mathscr{L}(E,M)$.

A set $Q \subseteq M_s$ is *equational* if it is equal to $\mathscr{L}(E,M).x_1$, where E is a system of equations.

Equational sets and systems have almost all the usual properties [10]. The only difference to the standard case is that the emptiness of a component $L(E,M).x_i$ cannot, in general, be decided by looking at a "finite image" of it. One has to find a locally finite magma A with $A_{\rho(x_i)} \neq \emptyset$ and a relational homomorphism $\sigma : M \to A$ with total σ_s. This is possible if M is total (take the terminal object in the category of total magmas for A). Otherwise we have the following counterexample.

Example 4.3 (*finite images*). Let $M_s = \mathbb{N}$ and $f_M : \mathfrak{P}(M_s) \times \mathfrak{P}(M_s) \to \mathfrak{P}(M_s)$, $f_M(\{n\}, \{n+1\}) = \{n\}$ and $f_M(\{m\}, \{n\}) = \emptyset$ otherwise. There is no relational homomorphism $\sigma : M \to A$, where A is a nontrivial locally finite magma.

The most important property is that $\mathscr{L}(E,M) = \bigcup_{v=0}^{\infty} E_M^v(\emptyset, \ldots, \emptyset)$, where the union is taken componentwise. It is computable (under natural effectivity constraints on E) if all components are finite. Least solutions are preserved by relational homomorphisms: If $\varphi : M \to M'$ is a relational homomorphism and E a system of equations then $\mathscr{L}(E,M') = \varphi(\mathscr{L}(E,M))$, where φ is taken componentwise.

Let us write $e_i = t_{i1} +_{s_i} \ldots +_{s_i} t_{ik_i}$ for each $i \in [n]$. A standard procedure [26] allows to make every system of equations *uniform*, such that every t_{ij} has the form $f x_{\mu_1} \ldots x_{\mu_l}$, $f \in \mathscr{F}$, $\mu_1, \ldots, \mu_l \in [n]$. Least solutions can be characterized via ground rewriting systems.

Definition 4.4 (*ground rewriting system*). Let E be a system of equations

$$x_1 = t_{11} +_{s_1} \ldots +_{s_1} t_{1k_1}, \quad \ldots, \quad x_n = t_{n1} +_{s_n} \ldots +_{s_n} t_{nk_n} .$$

The associated *ground rewriting system* consists of the rules $x_i \underset{\mathscr{E}}{\longrightarrow} t_{ij}$, $i \in [n]$, $j \in [k_i]$. It considers $x_i \in \mathscr{F} \cup \{x_1, \ldots, x_n\}$ as a constant instead of a variable.

Lemma 4.5 (ground rewriting system).

$$\mathscr{L}(E,M).x_i = \bigcup_{t \in \mathscr{T}_{Q(x_i)}(\mathscr{F}) \atop x_i \xrightarrow[\mathscr{E}]{+} t} M(t)\,,$$

where $\xrightarrow[\mathscr{E}]{}$ *is the ground rewriting relation induced by* E.

We omit the proof, which is a straightforward adaption of the proof in [10]. As a consequence, *derivation trees* can be defined in the usual sense. Finally, it is clear that the class of equational sets is closed under the operations Ω_s, $+_s$ and those of $\mathscr{F}$.

Example 4.6 (*equational sets*). Equational sets (using interesting operations) include

(1) equational sets in the usual sense (defined by polynomial systems);

(2) the set of unrooted trees (Courcelle). It is decribed by the equation

$$x = e + \mathrm{glue}(x,x)\,,$$

where the constant e denotes a singleton set of an edge (with two endpoints), and the binary operation glue identifies an arbitrary node of the first tree with an arbitrary node of the second. A different possibility is

$$x = 1 + \mathrm{connect}(x,x)\,,$$

where the constant 1 denotes a singleton set of a node, and the binary operation connect adds an edge between an arbitrary node of the first tree and an arbitrary node of the second.

(3) the set of chordal graphs (Courcelle). Although this set is not equational with respect to deterministic operations [7], it is described by the equation

$$x = 1 + \mathrm{extend}(x)\,,$$

where the constant 1 denotes a singleton set of a vertex, and the unary operation extend chooses a clique and adds edges between each vertex in it and a new vertex;

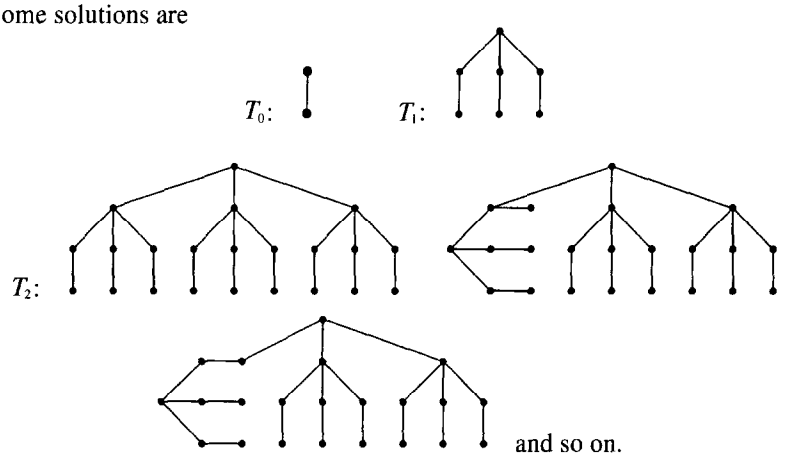
(4) the set of trees among the forbidden minors of the set of graphs of path-width at most n [29]. (This example was also suggested by Courcelle.) It is described by the first $n+1$ equations

$$T_0 = e$$

$$T_{i+1} = f(T_i, T_i, T_i)\,,$$

where the constant e denotes a singleton set of an edge (with two endpoints), and the operation f glues its arguments in (by identification of two vertices) at three places:

Some solutions are



5. Sets of Evaluations

Sets of evaluations generalize (relational) homomorphisms, context mappings and inductive sets of predicates at the same time. Another nice paradigm is to allow (non-deterministic) automata to produce output. Both views generalize different kinds of tree transducers. Such transformations are useful because they preserve equational sets. We introduce these devices and list their closure properties.

Definition 5.1 (*set of evaluations*). A *set of evaluations* $\mathscr{V}$ on a magma M in a magma M' is given by a mapping $\alpha\colon \mathscr{V} \to \mathscr{S}$ ($\alpha(v)$ is called the *sort* of v) and an interpretation $\hat{v}$ for every $v \in \mathscr{V}$ such that $\hat{v}\colon \mathfrak{P}(M_{\alpha(v)}) \to \mathfrak{P}(M'_{\alpha(v)})$ is a relational mapping. We assume that M, M' have respective signatures $(\mathscr{S},\mathscr{F})$, $(\mathscr{S}',\mathscr{F}')$ with $\mathscr{S} \subseteq \mathscr{S}'$.

A set of evaluations $\mathscr{V}$ is *locally finite* if $\mathscr{V}_s = \{v \in \mathscr{V} \mid \alpha(v) = s\}$ is finite for every $s \in \mathscr{S}$.

A set of evaluations $\mathscr{V}$ is $\mathscr{F}$-*inductive* if, for every $v \in \mathscr{V}_s$ and every $f \in \mathscr{F}$ of type $s_1 \dots s_n s$, there exist $k \in \mathbb{N}$, linear terms $t_i \in \mathscr{T}_s(\mathscr{F}', x_{i1}, \dots, x_{in})$, $\rho(x_{ij}) = s_j$, $i \in [k]$, $j \in [n]$, and $v_{ij} \in \mathscr{V}_{s_j}$, $i \in [k]$, $j \in [n]$, such that, for all $X_j \subseteq M_{s_j}$, $j \in [n]$:

$$\hat{v}(f_M(X_1, \dots, X_n)) = \bigcup_{i \in [k]} M'(t_i)(\hat{v}_{i1}(X_1), \dots, \hat{v}_{in}(X_n)) .$$

The sequence $t_1 v_{11} \dots v_{1n} \dots t_k v_{k1} \dots v_{kn}$ is called a *decomposition* of v relative to f.

An $\mathscr{F}$-inductive set of evaluations in M' is *total* if the mapping $M'(t_i)\colon \mathfrak{P}(M'_{s_1}) \times \dots \times \mathfrak{P}(M'_{s_n}) \to \mathfrak{P}(M'_s)$ is total for every $t_i \in \mathscr{T}_s(\mathscr{F}', x_{i1}, \dots, x_{in})$ appearing in the decomposition of some $v \in \mathscr{V}_s$ relative to some $f \in \mathscr{F}$ of type $s_1 \dots s_n s$.

Note that a set of evaluations in a total magma M' is total if no term t_i is a constant denoting the empty set.

Example 5.2 (*valid formulas*). A prominent evaluation assigns to a (simple) graph the set of CMSOL formulas it satisfies [4]. See Section 6 for a more precise statement. Also very useful is the evaluation which assigns to a (simple) graph the set of subgraphs satisfying a certain CMSOL formula [15]. These subgraphs can be mapped into a commutative semiring, leading to an extension of Parikh's theorem.

Proposition 5.3 (regular image 1). *If Q is equational then $\hat{v}(Q)$ is equational, where $v \in \mathcal{V}$ and $\mathcal{V}$ is a locally finite, $\mathcal{F}$-inductive set of evaluations on M in M'.*

We omit the proof, which is a straightforward adaption of the proof in [10]. It is similar to the proof of Proposition 5.6 below and in fact follows from it by Lemma 5.5 if $\mathcal{V}$ in M' is total.

The notion of a set of evaluations corresponds to what is usually called a *linear top-down* (*root-to-frontier*) *tree transducer*. Although it has proved to be very useful, *linear bottom-up* (*frontier-to-root*) *tree transducers* normally provide a more general class and possess better closure properties. (See also [20, 23, 24].) The additional expressive power is not an issue in most cases: Bottom-up tree transducers are able to examine subtrees which are discarded later. This can also be achieved by operations of a top-down transducer if the magma is not free. In fact, all transducers in the next section will have a rather simple structure. What is really needed is closure under composition.

Tree transducers are usually defined as term rewriting systems driven by tree automata. The rewriting system transforms an input term into an output term. The outputs generated from the subtrees $t_1, \ldots, t_n$ are combined according to the chosen transition of the automaton on f to yield the output for the whole tree $ft_1 \ldots t_n$. We prefer to use an equational notation instead of a rewriting system (in analogy to Lemma 4.5) to emphasize that our magmas are in most cases not free. Similarly to automata before, we introduce transducers in an algebraic version that works on all magmas M without the assumptions that they are total or generated by the initial relational homomorphism $\mathsf{val}: \mathcal{T}(\mathcal{F}) \to M$.

Definition 5.4 (*transducer, regular image*). Consider again two magmas M and M' with respective signatures $(\mathcal{S}, \mathcal{F})$ and $(\mathcal{S}', \mathcal{F}')$, where, for sake of simplicity, $\mathcal{S} \subseteq \mathcal{S}'$. A *transducer* from M to M' consists of a (nondeterministic) automaton (σ, A, F) on M and a mapping d such that $d(f, a, a_1, \ldots, a_n) \in \mathcal{E}_s(\mathcal{F}', x_1, \ldots, x_n)$, where $f \in \mathcal{F}$ has type $s_1 \ldots s_n s$, $a \in A_s$, $a_i \in A_{s_i}$, $\rho(x_i) = s_i$, $i \in [n]$, and the terms in $d(f, a, a_1, \ldots, a_n)$ are linear.

A transducer is *deterministic* if the automaton (σ, A, F) is deterministic and $d(f, a, a_1, \ldots, a_n) \in \mathcal{T}_s(\mathcal{F}', x_1, \ldots, x_n)$ for all $f \in \mathcal{F}$ and $a \in A_s$, $a_i \in A_{s_i}$, $i \in [n]$.

A set of evaluations $\mathscr{V}$ is *compatible* with a transducer if $\mathscr{V}_s = A_s$ and, for every $a \in A_s$, for every $f \in \mathscr{F}$ of type $s_1 \dots s_n s$ and for all $X_i \subseteq M_{s_i}$, $i \in [n]$,

$$\hat{a}(f_M(X_1,\dots,X_n)) = \bigcup_{\substack{a_1 \in A_{s_1},\dots,a_n \in A_{s_n} \\ a \in f_A(\{a_1\},\dots,\{a_n\})}} M'(d(f,a,a_1,\dots,a_n))(\hat{a}_1(X_1),\dots,\hat{a}_n(X_n)) ;$$

moreover, $\hat{a}(\{m \in M_s \mid a \notin \sigma_s(\{m\})\}) = \emptyset$.

The *regular image* of a set $X \subseteq M_s$ under a compatible set of evaluations is $\bigcup_{a \in F} \hat{a}(X)$.

A compatible set of evaluations is *total* if the mapping

$$M'(d(f,a,a_1,\dots,a_n)) : \mathfrak{P}(M'_{s_1}) \times \cdots \times \mathfrak{P}(M'_{s_n}) \to \mathfrak{P}(M'_s)$$

is total for every $d(f,a,a_1,\dots,a_n) \in \mathscr{E}_s(\mathscr{F}',x_1,\dots,x_n) \setminus \{\Omega_s\}$, $f \in \mathscr{F}$ of type $s_1 \dots s_n s$, $a \in A_s$, $a_i \in A_{s_i}$, $i \in [n]$.

Intuitively, in a compatible set, the evaluations are associated with states of an automaton. No output is produced if the automaton does not take on the required state on its input. (This holds automatically if the magma M is generated by the initial relational homomorphism $\mathsf{val}: \mathscr{T}(\mathscr{F}) \to M$.) The mapping d associates an output with a transition of the automaton. The overall result is valid only if the automaton ends up in an accepting state. Before we prove that equational sets are preserved under regular images, we show that, as announced, most $\mathscr{F}$-inductive sets of evaluations are instances of the preceding definition.

Lemma 5.5 (inductive set). *The image $\hat{v}(X)$ of a set X under $v \in \mathscr{V}$ is a regular image, where $\mathscr{V}$ is a total, locally finite, $\mathscr{F}$-inductive set of evaluations on a magma M in a magma M'.*

Proof. We claim that $\mathscr{V}$ is compatible with the following transducer from M to M':

- $A_s = \mathscr{V}_s$ for every $s \in \mathscr{S}$,

$$f_A(\{a_1\},\dots,\{a_n\}) = \{a \in A_s \mid a_1 \dots a_n \text{ is part of the decomposition of } a \text{ relative to } f\}$$

  for every $f \in \mathscr{F}$ of type $s_1 \dots s_n s$ and for all $a_i \in A_{s_i}$, $i \in [n]$.
- $\sigma_s(\{m\}) = \{a \in A_s \mid \hat{a}(\{m\}) \neq \emptyset\}$ for all $m \in M_s$ and for every $s \in \mathscr{S}$.
- $F = \{v\}$.
- $d(f,a,a_1,\dots,a_n) = \sum_{ta_1\dots a_n \text{ is part of the decomposition of } a \text{ relative to } f} t$

Obviously, (σ,A,F) is a (nondeterministic) automaton on M, and the expressions $d(f,a,a_1,\dots,a_n)$ simply repeat the original induction schemes. □

Proposition 5.6 (regular image 2). *If Q is equational then the regular image of Q under the set of evaluations $\mathscr{V}$ is equational, where $\mathscr{V}$ is compatible with the transducer (σ,A,F,d).*

Proof. We have $Q = \mathscr{L}(E,M).x_1$, where E is a uniform system of equations. We define a new system of equations E' with variables $[x_i,a]$, $i \in [n]$, $a \in A_{\rho(x_i)}$, such that $\mathscr{L}(E',M').[x_i,a] = \hat{a}(\mathscr{L}(E,M).x_i)$. Here is one equation of E':

$$[x_i,a] = \ldots + \sum_{\substack{a_1 \in A_{s_1},\ldots,a_m \in A_{s_m} \\ a \in f_A(\{a_1\},\ldots,\{a_m\})}} d(f_{ij},a,a_1,\ldots,a_m)([x_{l_1},a_{l_1}],\ldots,[x_{l_m},a_{l_m}]) + \ldots ,$$

where f_{ij} belongs to the jth monomial in the righthand side of the ith equation.

We prove $(\hat{a}(E_M^v(\emptyset,\ldots,\emptyset).x_i) = E_{M'}'^v(\emptyset,\ldots,\emptyset).[x_i,a])$ by induction on v. $v = 0$: $(\hat{a}(E_M^0(\emptyset,\ldots,\emptyset).x_i) = \emptyset = E_{M'}'^0(\emptyset,\ldots,\emptyset).[x_i,a])$. $v \to v+1$:

$$\begin{aligned}
&\hat{a}(E_M^{v+1}(\emptyset,\ldots,\emptyset).x_i) \\
&= \hat{a}(\bigcup_j f_{ij_M}(E_M^v(\emptyset,\ldots,\emptyset))) = \bigcup_j \hat{a}(f_{ij_M}(E_M^v(\emptyset,\ldots,\emptyset))) \\
&= \bigcup_j \bigcup_{\substack{a_1 \in A_{s_1},\ldots,a_m \in A_{s_m} \\ a \in f_A(\{a_1\},\ldots,\{a_m\})}} M'(d(f_{ij},a,a_1,\ldots,a_m))(\hat{a}_{l_1}(E_M^v(\emptyset,\ldots,\emptyset).x_{l_1}),\ldots, \\
&\qquad\qquad \hat{a}_{l_m}(E_M^v(\emptyset,\ldots,\emptyset).x_{l_m})) \\
&= \bigcup_j \bigcup_{\substack{a_1 \in A_{s_1},\ldots,a_m \in A_{s_m} \\ a \in f_A(\{a_1\},\ldots,\{a_m\})}} M'(d(f_{ij},a,a_1,\ldots,a_m))(E_{M'}'^v(\emptyset,\ldots,\emptyset).[x_{l_1},a_{l_1}],\ldots, \\
&\qquad\qquad E_{M'}'^v(\emptyset,\ldots,\emptyset).[x_{l_m},a_{l_m}]) \\
&= E_{M'}'^{v+1}(\emptyset,\ldots,\emptyset).[x_i,a] .
\end{aligned}$$

Now we introduce a new variable x and the equation

$$x = \sum_{a \in F} [x_1,a] .$$

Then the regular image of Q under $\mathscr{V}$ is $\bigcup_{a \in F} \hat{a}(\mathscr{L}(E,M).x_1) = \mathscr{L}(E',M').x$. □

The following laborious propositions and lemmas establish the important closure properties for the class of compatible sets of evaluations. An interesting result is Corollary 5.11 which gives a short proof for the well-known and useful fact that intersection with a deterministically recognizable set preserves equational sets.

Proposition 5.7 (closure properties 1). *If the sets of evaluations $\mathscr{V}$ and $\mathscr{V}'$ are both compatible with transducers (σ,A,F,d) from M to M' and (σ',A',F',d') from M' to M'', respectively, then the set of evaluations $\mathscr{V}''$ defined by*

$$\mathscr{V}_s'' = \{v' \circ v \mid v \in \mathscr{V}_s, v' \in \mathscr{V}_s'\}$$

for every $s \in \mathscr{S}$ is compatible with a transducer from M to M''. The transducer is deterministic if both, (σ,A,F,d) and (σ',A',F',d') are deterministic and $\sigma_s' \circ \hat{a} \restriction \bar{\sigma}_s^{-1}(\{a\})$ is functional for every $a \in A_s$, $s \in \mathscr{S}$.

Proof. We first define $d'(e,a,a_1,\ldots,a_n)$ inductively for general expressions $e \in \mathscr{E}_s(\mathscr{F}',x_1,\ldots,x_n)$ containing only linear terms:

- $e = t_1 +_s \ldots +_s t_m$:

$$d'(t_1 +_s \ldots +_s t_m, a, a_1, \ldots, a_n) = d'(t_1, a, a_1, \ldots, a_n) +_s \ldots +_s d'(t_m, a, a_1, \ldots, a_n) .$$

  In particular, $d'(\Omega_s, a, a_1, \ldots, a_n) = \Omega_s$ for $m = 0$.
- $e = x_i$: $d'(x_i, a, a_1, \ldots, a_n) = x_i$, $i \in [n]$.
- $e = f t_1 \ldots t_m$: Assume that t_i, $i \in [m]$, contains the variables $x_{j_{i1}}, \ldots, x_{j_{ik_i}}$, $j_{i1}, \ldots, j_{ik_i} \in [n]$ (the variables $x_1, \ldots, x_n$ can be partitioned among the terms t_i), and that $d'(t_i, a', a_{j_{i1}}, \ldots, a_{j_{ik_i}}) = \ldots +_{s_i} t^{(i)}_{a'l} +_{s_i} \ldots$ for $a' \in f_{A'}(\{a_{j_{i1}}\}, \ldots, \{a_{j_{ik_i}}\})$. (We apologize for the many subscripts.) Then

$$d'(f t_1 \ldots t_m, a, a_1, \ldots, a_n)$$

$$= \sum_{\substack{a'_1 \in f_{A'}(\{a_{j_{11}}\}, \ldots, \{a_{j_{1k_1}}\}) \\ \ldots \\ a'_m \in f_{A'}(\{a_{j_{m1}}\}, \ldots, \{a_{j_{mk_m}}\}) \\ a \in f_{A'}(\{a'_1\}, \ldots, \{a'_m\})}} \sum_{i_1, \ldots, i_m} d'(f, a, a'_1, \ldots, a'_m)[x_1 := t^{(1)}_{a'_1 i_1}, \ldots, x_m := t^{(m)}_{a'_m i_m}] .$$

  (It is possible that $d'(f t_1 \ldots t_m, a, a_1, \ldots, a_n) = \Omega_s$.) It follows that

$$\hat{a}(M'(e)(X_1, \ldots, X_n)) = \bigcup_{\substack{a_1 \in A'_{s_1}, \ldots, a_n \in A'_{s_n} \\ a \in A'(e)(\{a_1\}, \ldots, \{a_n\})}} M''(d'(e, a, a_1, \ldots, a_n))(\hat{a}_1(X_1), \ldots, \hat{a}_n(X_n))$$

  for all $a \in A'_s$ and all $X_i \subseteq M'_{s_i}$, $i \in [n]$. With this preparation, a transducer $(\sigma'', A'', F'', d'')$ for $\mathscr{V}''$ can be given as follows:
- $A''_s = A_s \times A'_s$ for every $s \in \mathscr{S}$, $\widehat{\langle a, a' \rangle} = \hat{a}' \circ \hat{a}$,

$$f_{A''}(\{\langle a_1, a'_1 \rangle\}, \ldots, \{\langle a_n, a'_n \rangle\})$$

$$= \bigcup_{a \in f_A(\{a_1\}, \ldots, \{a_n\})} \{a\} \times A'(d(f, a, a_1, \ldots, a_n))(\{a'_1\}, \ldots, \{a'_n\})$$

  for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all $\langle a_i, a'_i \rangle \in A''_{s_i}$, $i \in [n]$.
- $\sigma''_s(X) = \bigcup_{a \in \sigma_s(X)} \{a\} \times \sigma'_s(\hat{a}(X))$ for all $X \subseteq M_s$ and every $s \in \mathscr{S}$.
- $F'' = F \times F'$.
- $d''(f, \langle a, a' \rangle, \langle a_1, a'_1 \rangle, \ldots, \langle a_n, a'_n \rangle) = d'(d(f, a, a_1, \ldots, a_n), a', a'_1, \ldots, a'_n)$.

We leave the tedious verifications to the reader. The statement for deterministic transducers follows easily. □

Lemma 5.8 (closure properties 2). *If the set of evaluations $\mathscr{V}$ is total and compatible with a transducer (σ, A, F, d) from M to M', then the set of all $m \in M_s$ which have a nonempty image under $\mathscr{V}$ is $\exists$-recognizable. It is deterministically recognizable if the transducer is deterministic.*

Proof. Because $\mathscr{V}$ in M' is total, the Boolean mapping

$$e(f, a, a_1, \ldots, a_n): (X_1 \neq \emptyset, \ldots, X_n \neq \emptyset) \mapsto M'(d(f, a, a_1, \ldots, a_n))(X_1, \ldots, X_n) \neq \emptyset$$

is well defined for $X_i \subseteq M_{s_i}$, $i \in [n]$. We construct a new automaton (σ', A', F') on M as follows:

- $A'_s = A_s = \mathscr{V}_s$, $s \in \mathscr{S}$,

$$f_{A'}: (X_1, \ldots, X_n) \mapsto$$

$$\{a \in A_s \mid (\exists a_1 \in A_{s_1}) \ldots (\exists a_n \in A_{s_n})$$

$$(a \in f_A(\{a_1\}, \ldots, \{a_n\}) \wedge e(f, a, a_1, \ldots, a_n)(a_1 \in X_1, \ldots, a_n \in X_n))\}$$

  for every $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$ and for all $X_i \subseteq A'_{s_i}$, $i \in [n]$.
- $\sigma'_s(X) = \{a \in A_s \mid \hat{a}(X) \neq \emptyset\}$, $s \in \mathscr{S}$.
- $F' = F$.

It is easy to verify that σ' is a relational homomorphism. The set in question is equal to

$$\{m \in M_s \mid \sigma'_s(\{m\}) \cap F' \neq \emptyset\},$$

hence $\exists$-recognizable. The statement for a deterministic transducer follows easily. □

It follows that the set of all $m \in M_s$ whose image under $\mathscr{V}$ is empty is $\forall$-recognizable.

Lemma 5.9 (closure properties 3). *If the sets of evaluations $\mathscr{V}$ and $\mathscr{V}'$ are both compatible with transducers (σ, A, F, d) and (σ', A', F', d') from M to M', then the set of evaluations $\mathscr{V}''$ defined by*

$$\mathscr{V}''_s = \{v'': X \mapsto v(X) \cup v'(X) \mid v \in \mathscr{V}_s, v' \in \mathscr{V}'_s\}$$

for every $s \in \mathscr{S}$ is compatible with a transducer from M to M'.

Proof. A transducer for $\mathscr{V}''$ is given by the product automaton to (σ, A, F) and (σ', A', F') in the relational category, namely $(\sigma'', A \times A', F \uplus F')$ where σ'' is the induced relational homomorphism. Since mixed arguments are excluded, d and d' simply coexist. □

Lemma 5.10 (closure properties 4). *Relational homomorphisms and intersections with deterministically recognizable sets are sets of evaluations which are compatible with deterministic transducers.*

Proof. A relational homomorphism $\psi: M \to M'$ with distinguished component ψ_r, $r \in \mathscr{S}$, is obviously compatible with the deterministic transducer $(\sigma, A, \{r\}, d)$, where A is the terminal object (with $A_s = \{s\}$, $s \in \mathscr{S}$) and σ the induced homomorphism, $\hat{s} = \psi_s$, $s \in \mathscr{S}$, and $d(f, s, s_1, \ldots, s_n) = f x_1 \ldots x_n$ for $s, s_1, \ldots, s_n \in \mathscr{S}$, $f \in \mathscr{F}$ of type $s_1 \ldots s_n s$.

Secondly, let the deterministic automaton (σ, A, F) recognize R. We extend it to a deterministic transducer from M to M by $\hat{a}(X) = X \cap \bar{\sigma}_s^{-1}(\{a\})$ for all $X \subseteq M_s$, $a \in A_s$, $s \in \mathscr{S}$, and $d(f, a, a_1, \ldots, a_n) = fx_1 \ldots x_n$. We verify:

$$f_M(X_1, \ldots, X_n) \cap \bar{\sigma}_s^{-1}(\{a\}) = \bigcup_{\substack{a_1 \in A_{s_1}, \ldots, a_n \in A_{s_n} \\ f_A(a_1, \ldots, a_n) = a}} f_M(X_1 \cap \bar{\sigma}_{s_1}^{-1}(\{a_1\}), \ldots, X_n \cap \bar{\sigma}_{s_n}^{-1}(\{a_n\})) .$$

Then, $X \cap R = \bigcup_{a \in F} \hat{a}(X)$. □

Corollary 5.11 (intersection). *If $R \subseteq M_r$ is deterministically recognizable and $Q \subseteq M_r$ is equational then $Q \cap R$ is equational.*

Direct proofs for a corresponding result can be found in [10, 22].

Proof. The assertion follows from Proposition 5.6 and the preceding lemma. □

The assumption of deterministic recognizability in the preceding lemma and corollary cannot be weakened, as is shown by the following counterexample.

Example 5.12 (*intersection*). We extend the second item of Example 3.12. Let us consider a magma M over the signature $\mathscr{S} = \{s\}$, $\mathscr{F} = \{c, f\}$ with c of type s and f of type ss. It is given by $M_s = \mathbb{N}$, $c_M = \{0\}$ and $f_M : \{n\} \mapsto \{n, n+1\}$. The same automaton as before (extended by $c_A = \{0\}$) can be used to show that the sets $R_k = \{n \in \mathbb{N} \mid n \geqslant k\}$, $k \geqslant 0$, are $\exists$-recognizable and $\forall$-recognizable. The set $Q = \mathbb{N}$ is equational (it is described by the equation $x = c + fx$), but $R_k = Q \cap R_k$ is not if $k \neq 0$. Assume the contrary: $R_k = \mathscr{L}(E, M).x_1$ for some system of equations E. It is easy to prove by induction on v that every component of $E_M^v(\emptyset, \ldots, \emptyset)$ is either empty or contains 0. The same statement holds for $\mathscr{L}(E, M) = \bigcup_{v=0}^{\infty} E_M^v(\emptyset, \ldots, \emptyset)$, a contradiction.

Note that when looking at an equational set $Q \subseteq M_s$ one can always assume that the initial relational homomorphism $\mathsf{val} : \mathscr{T}(\mathscr{F}) \to M$ generates M. Lemma 5.10 allows to decide the emptiness of $Q \cap R$ by looking at its finite image in the automaton. The following lemma makes a weaker statement for intersections with nondeterministically recognizable sets.

Lemma 5.13 (decidability). *The following questions are decidable under natural effectivity constraints on the (nondeterministic) automaton and the system of equations:*

- *If $R \subseteq M_s$ is $\exists$-recognizable by a (nondeterministic) automaton and $Q \subseteq M_s$ is equational, is $Q \cap R \neq \emptyset$?*
- *If $R \subseteq M_s$ is $\forall$-recognizable by a (nondeterministic) automaton and $Q \subseteq M_s$ is equational, is $Q \subseteq R$?*

Proof. Let (σ, A, F) be a (nondeterministic) automaton for R and let E be a system of equations such that $Q = \mathscr{L}(E, M).x_1$. We show the first part, the second

being analogous. If $m \in Q \cap R = \mathscr{L}(E,M).x_1 \cap \{m \in M_s \mid \sigma_s(\{m\}) \cap F \neq \emptyset\}$ then $\sigma_s(\{m\}) \subseteq \mathscr{L}(E,A).x_1$ and $\sigma_s(\{m\}) \cap F \neq \emptyset$. Therefore, $\mathscr{L}(E,A).x_1 \cap F \neq \emptyset$. On the other hand, if $a \in \mathscr{L}(E,A).x_1 \cap F$, there exists $m \in \mathscr{L}(E,M).x_1 = Q$ such that $a \in \sigma(\{m\})$. We also have $m \in R$ since $a \in F$. But the condition $\mathscr{L}(E,A).x_1 \cap F \neq \emptyset$ is decidable. □

6. Finite relational structures, deterministic operations and counting monadic second-order logic

We apply the theory to magmas which contain finite relational structures as objects and suitably defined operations on them. The transducers of the preceding section yield transformations of finite relational structures on a syntactic level, based on their representation by terms. We also recall counting monadic second-order logic which is able to express properties of such structures. Definable transductions transform finite relational structures on a semantical level, only regarding their logical properties. Our main theorem states that all semantical transformations can be achieved by syntactic transformations.

6.1. Deterministic operations on finite relational structures

We start by introducing finite relational structures as objects. They also rely on a signature and an interpretation for it. In order to keep the levels apart, we confine ourselves with only one sort, which we do not need to mention. The signatures here contain a finite set of relation symbols and a finite set of constants instead of arbitrarily many operation symbols. This should make it sufficiently hard to confuse signatures for finite relational structures with signatures for magmas. Finite relational structures are objects of magmas (with operations to be defined later) and, therefore, a signature of a finite relational structure serves as a sort in the containing magma.

Definition 6.1 (*signature*). A *signature* consists of a finite set $\mathscr{R}$ of *relations* together with a mapping $\alpha: \mathscr{R} \to \mathbb{N}_+$, which gives the *arity* of $r \in \mathscr{R}$, and a finite set $\mathscr{C}$ of *constants*.

Unlike the situation for magmas, we will work with relational structures over different signatures. We therefore have to pay attention to $\mathscr{R}$ and $\mathscr{C}$.

Definition 6.2 (*finite relational structure*). A *finite relational structure* S consists of a finite set D_S (the *domain*), a relation $r_S \subseteq D_S^{\alpha(r)}$ for every $r \in \mathscr{R}$, and an element $c_S \in D_S$ for every $c \in \mathscr{C}$.

The class of finite relational structures over $\mathscr{R}$ and up to isomorphism is written as $\mathfrak{S}(\mathscr{R},\mathscr{C})$.

Constants play an essential role in the gluing operation (parallel composition) below. They make a difference (which does not change the expressive power with respect

to CMSOL) in the quantifier-free definable operations. But they are inconvenient in definable transductions, also introduced later. Therefore, we sometimes prefer to regard $S \in \mathfrak{S}(\mathscr{R},\mathscr{C})$ as a finite relational structure over the signature $\mathscr{R} \cup \mathscr{C}$ without constants. A $c \in \mathscr{C}$ is then interpreted as a unary relation $c_S \subseteq D_S$ containing exactly one element. We set $\alpha(c) = 1$ accordingly.

Our main examples of finite relational structures are labelled, directed hypergraphs with sources. We simply call them *graphs* in the rest of this article. We define graphs and simple graphs below. Also, hyperedges are simply called *edges*. Let us fix a (sufficiently large) finite set L of *labels* together with a *rank* mapping $\rho : L \to \mathbb{N}_+$.

In a graph, the edges carry a label from L and there are $n \geqslant 0$ distinguished vertices.

Definition 6.3 (*graph*). A *graph* G of *sort* $n \geqslant 0$ consists of
- a finite set V of *vertices*,
- a finite set E of *edges*,
- a *vertex* mapping $\mathrm{vert} : E \to V^*$,
- a *label* mapping $\mathrm{lab} : E \to L$ and
- a *source* mapping $\mathrm{src} : [n] \to V$,

such that $|\mathrm{vert}(e)| = \rho(\mathrm{lab}(e))$ for all $e \in E$. The vertex $\mathrm{src}(n)$ is called the nth source. We assume that $V \cap E = \emptyset$.

Example 6.4 (*graphs*). A finite relational structure for graphs of sort n contains the following:
- the domain $D = V \cup E$,
- a relation edge_l of arity $\rho(l) + 1$ for every $l \in L$ such that

$$\mathrm{edge}_l(e, v_1, \ldots, v_n) \iff \mathrm{lab}(e) = l \wedge \mathrm{vert}(e) = v_1 \ldots v_n$$

  (i.e., $e \in E$ and $v_1, \ldots, v_n \in V$),
- a constant s_i for every $i \in [n]$ such that $s_i = \mathrm{src}(i)$.

The additional constraint that no two different tuples in the relation $\bigcup_{l \in L} \mathrm{edge}_l$ share their first component is not expressed directly.

In a simple graph, in contrast to the above, we do not distinguish between several edges connecting the same vertices and carrying the same label.

Definition 6.5 (*simple graph*). A *simple graph* G of *sort* $n \geqslant 0$ consists of
- a finite set V of *vertices*,
- a finite set $E \subseteq L \times V^*$ of *edges*,
- a *source* mapping $\mathrm{src} : [n] \to V$,

such that $|\vec{v}| = \rho(l)$ for all $(l, \vec{v}) \in E$.

Example 6.6 (*simple graphs*). A finite relational structure for simple graphs of sort n contains the following:

- the domain $D = V$,
- a relation edge'_l of arity $\rho(l)$ for every $l \in L$ such that

$$\text{edge}'_l(v_1,\ldots,v_n) \iff (l, v_1 \ldots v_n) \in E\,,$$

- a constant s_i for every $i \in [n]$ such that $s_i = \text{src}(i)$.

Simple graphs of sort n are indeed finite relational structures over the signature $(L,[n])$.

We now define operations on isomorphism classes of finite relational structures according to [6]. As usual, the symbol // (and, strictly speaking, also the symbol def_Δ for a given Δ) is overloaded, i.e., we use the same symbol for "similar" operations of different types.

Definition 6.7 (*parallel composition*). *Parallel composition* // is a binary gluing operation on finite relational structures. Let $(S_1 \in \mathfrak{S}(\mathscr{R}_1,\mathscr{C}_1)),(S_2 \in \mathfrak{S}(\mathscr{R}_2,\mathscr{C}_2))$. We assume that D_{S_1} and D_{S_2} are disjoint. (There is no loss of generality because operations are defined up to isomorphism.) Let $\approx$ be the least equivalence relation on $D_{S_1} \cup D_{S_2}$ such that $c_{S_1} \approx c_{S_2}$ for every $c \in \mathscr{C}_1 \cap \mathscr{C}_2$. Then, $S_1//S_2$ is the following structure $T \in \mathfrak{S}(\mathscr{R}_1 \cup \mathscr{R}_2, \mathscr{C}_1 \cup \mathscr{C}_2)$:

- $D_T = (D_{S_1} \cup D_{S_2})/\approx$,
- $r_T([d_1]_\approx,\ldots,[d_{\alpha(r)}]_\approx)$ holds if $r_{S_1}(d'_1,\ldots,d'_{\alpha(r)})$ holds or if $r_{S_2}(d'_1,\ldots,d'_{\alpha(r)})$ holds for some $d'_1 \in [d_1]_\approx,\ldots,d'_{\alpha(r)} \in [d_{\alpha(r)}]_\approx$,
- $c_T = [c_{S_1}]_\approx$ if $c \in \mathscr{C}_1$ and $c_T = [c_{S_2}]_\approx$ if $c \in \mathscr{C}_2$.

Parallel composition is associative and commutative. It forms the coproduct in the category of finite relational structures over a given signature.

Let $\vec{x}$ be a finite sequence (without repetitions) of *object variables*. We denote by $\text{QF}(\mathscr{R},\mathscr{C},\vec{x})$ the set of formulas written with predicate symbols $\mathscr{R}$, constants $\mathscr{C}$, variables in $\vec{x}$ and the Boolean connectives $\neg$, $\wedge$, $\vee$ etc.

Definition 6.8. [quantifier-free definable operations] A *quantifier-free definition scheme* Δ from $(\mathscr{R},\mathscr{C})$ to $(\mathscr{R}',\mathscr{C}')$ is specified by

- $\psi \in \text{QF}(\mathscr{R},\mathscr{C},x_1)$ of the form $\psi' \vee \bigvee_{c\in\mathscr{C}'} x_1 = \kappa_c$, describing the domain of the target structure,
- $\vartheta_r \in \text{QF}(\mathscr{R},\mathscr{C},x_1 \ldots x_{\alpha(r)})$, $r \in \mathscr{R}'$, describing the relations in the target structure, and
- $\kappa_c \in \mathscr{C}$, $c \in \mathscr{C}'$, giving the constants.

Δ determines a mapping $\text{def}_\Delta\colon \mathfrak{S}(\mathscr{R},\mathscr{C}) \to \mathfrak{S}(\mathscr{R}',\mathscr{C}')$, which is called *quantifier-free definable*. Let $S \in \mathfrak{S}(\mathscr{R},\mathscr{C})$. $T = \text{def}_\Delta(S)$ is the following structure:

- $D_T = \{d \in D_S \mid S \models \psi(d)\}$,
- $r_T(d_1,\ldots,d_{\alpha(r)})$ holds if $d_1,\ldots,d_{\alpha(r)} \in D_T$ and $S \models \vartheta_r(d_1,\ldots,d_{\alpha(r)})$ for every $r \in \mathscr{R}'$,
- c_T is κ_{cS} for every $c \in \mathscr{C}'$.

Definition 6.9 (*constants*). The constants $0_{\mathscr{R},\mathscr{C}}$ and $1_{\mathscr{R},\mathscr{C}}$ denote finite relational structures over the signature $(\mathscr{R},\mathscr{C})$. $0_{\mathscr{R},\mathscr{C}}$ denotes the discrete structure consisting of domain $P\mathscr{C}$, empty relations and the obvious constants. $1_{\mathscr{R},\mathscr{C}}$ consists of a one-element domain and one-element relations.

$0_{\emptyset,\emptyset}$ is the neutral element with respect to //. $0_{\mathscr{R},\mathscr{C}}$ and $1_{\mathscr{R},\mathscr{C}}$ are the respective initial and terminal objects in the category of finite relational structures over the signature $(\mathscr{R},\mathscr{C})$.

The constants $0_{\mathscr{R},\mathscr{C}}$ for $\mathscr{C} \neq \emptyset$ are not needed because they can be derived from the other operations. In this section, all signatures will be taken relative to a base signature $(\mathscr{R},\mathscr{C})$. $\mathscr{F}_{\mathscr{R},\mathscr{C}}$ refers to the set of operations containing $0_{\mathscr{R},\emptyset}$, $1_{\mathscr{R},C}$, $C \subseteq \mathscr{C}$, all quantifier-free definable operations from $(\mathscr{R},C)$ to $(\mathscr{R},C')$, $C,C' \subseteq \mathscr{C}$, and // with arguments over $(\mathscr{R},C_1)$ and $(\mathscr{R},C_2)$, $C_1,C_2 \subseteq \mathscr{C}$. The corresponding set of sorts would be $\mathscr{S}_{\mathscr{R},\mathscr{C}} = \{(\mathscr{R},C) \mid C \subseteq \mathscr{C}\}$. The operations just introduced turn $\bigcup_{C \subseteq \mathscr{C}} \mathfrak{S}(\mathscr{R},C)$ into a functional magma $\mathscr{M}_{\mathscr{R},\mathscr{C}}$.

Remark 6.10. We fix the set of relations $\mathscr{R}$ and let the set of constants $C \subseteq \mathscr{C}$ vary because relations can be empty. A structure with empty relations is identified with a structure that does not contain these relations at all. By this convention, we can avoid to introduce (quantifier-free definable) operations to forget relations. This corresponds to the intuition behind (simple) graphs, where the set of labels L is likewise assumed to be fixed.

Deterministic operations along these lines were introduced in [2] for graphs and in [6] for simple graphs, the latter being more general than those in [14]. They are all derived from the operations in $\mathscr{F}_{\mathscr{R},\mathscr{C}}$, where $\mathscr{R}$ is L (more exactly, $\mathscr{R} = \{\mathrm{edge}_l \mid l \in L\}$ or $\mathscr{R} = \{\mathrm{edge}'_l \mid l \in L\}$, respectively) and $\mathscr{C}$ is $[n]$ (more exactly, $\mathscr{C} = \{s_i \mid i \in [n]\}$), $n \in \mathbb{N}$. We list those involving quantifier-free definable operations and modify them slightly. The operation symbols are again overloaded.

The first group of operations corresponds to graph rewriting by *hyperedge replacement* (HR). In this approach, a graph is transformed by substituting another graph for an edge. The sources of the inserted graph are glued with the former attachment points of the edge which it replaces. The second group of operations roughly corresponds to graph rewriting by *vertex replacement* (VR). Here, a vertex (and all its adjacent edges) is replaced with another graph. The inserted graph is embedded by adding new edges to the former neighbours of the vertex. These edges and their labels further depend on the labels of the vertices involved and the removed edges. We refer the reader to [6, 21, 22, 12] for a comparison between both approaches of graph rewriting, in particular on the algebraic and logical level.

Definition 6.11 (*HR operations*). The following *HR operations* are defined on graphs.

- $l \in L$ is a constant of type $\rho(l)$. It denotes the graph $G = (V,E,\mathrm{vert},\mathrm{lab},\mathrm{src})$ with one l-labelled edge together with $n = \rho(l)$ vertices, which are also the sources: $V = [n]$, $E = \{e\}$, $\mathrm{vert}(e) = 1 \ldots n$, $\mathrm{lab}(e) = l$ and $\mathrm{src} = \mathrm{id}_{[n]}$.

- ren_f is a unary operation of type mn, where $f:[n]\to[m]$ is a mapping. Let $G=(V,E,\mathrm{vert},\mathrm{lab},\mathrm{src})$ be a graph of sort m. Then $\mathrm{ren}_f(G)$ denotes the graph $G'=(V',E',\mathrm{vert}',\mathrm{lab}',\mathrm{src}')$ with sources renamed: $V'=V$, $E'=E$, $\mathrm{vert}'=\mathrm{vert}$, $\mathrm{lab}'=\mathrm{lab}$ and $\mathrm{src}'=\mathrm{src}\circ f$. (Note that every constant $1_{L,[n]}$ is equal to $\mathrm{ren}_f(1_{L,[1]})$, where $f:[n]\to[1]$.)
- fuse_δ is a unary operation of type nn, where $\delta\subseteq[n]\times[n]$ is an equivalence relation. Let $G=(V,E,\mathrm{vert},\mathrm{lab},\mathrm{src})$ be a graph of type n and let $\approx$ denote the smallest equivalence relation on V such that $v_1\approx v_2$ if $v_1=\mathrm{src}(i_1)$ and $v_2=\mathrm{src}(i_2)$ for some $i_1\ \delta\ i_2$. Then $\mathrm{fuse}_\delta(G)$ denotes the graph $G'=(V',E',\mathrm{vert}',\mathrm{lab}',\mathrm{src}')$ with glued vertices: $V'=V/{\approx}$, $E'=E$, $\mathrm{vert}'=[\ ]^*_\approx\circ\mathrm{vert}$, $\mathrm{lab}'=\mathrm{lab}$ and $\mathrm{src}'=[\ ]_\approx\circ\mathrm{src}$.
- $\oplus$ is a binary operation of type $n_1n_2(n_1+n_2)$. Let the arguments $G_1=(V_1,E_1,\mathrm{vert}_1,\mathrm{lab}_1,\mathrm{src}_1)$ and $G_2=(V_2,E_2,\mathrm{vert}_2,\mathrm{lab}_2,\mathrm{src}_2)$ be graphs of respective types n_1 and n_2 such that $V_1\cap V_2=\emptyset$ and $E_1\cap E_2=\emptyset$. Then $G_1\oplus G_2$ denotes the (disjoint) union $G=(V,E,\mathrm{vert},\mathrm{lab},\mathrm{src})$ of G_1 and G_2 with sources renamed in G_2: $V=V_1\cup V_2$, $E=E_1\cup E_2$, $\mathrm{vert}=\mathrm{vert}_1\cup\mathrm{vert}_2$, $\mathrm{lab}=\mathrm{lab}_1\cup\mathrm{lab}_2$ and $\mathrm{src}=\mathrm{src}_1\cup\mathrm{src}_2\circ\mathrm{sub}_{n_1n_2}$, where $\mathrm{sub}_{n_1n_2}(i)=i-n_1$ for $i\in[n_1+n_2]\setminus[n_1]$.

Definition 6.12 (*VR operations*). The following *VR operations* are defined on simple graphs. They are instances of quantifier-free definable operations.

- relab_r is a unary operation of type nn, where r is a *rank-preserving* relation, i.e., $r\subseteq\{(l_1,l_2)\in L\times L\mid\rho(l_1)=\rho(l_2)\}$. If $G=(V,E,\mathrm{src})$ is a simple graph, $\mathrm{relab}_r(G)$ denotes the simple graph $G'=(V',E',\mathrm{src}')$ with edges renamed by r: $V'=V$, $E'=\{(l',\vec{v})\mid(l,\vec{v})\in E,l\ r\ l'\}$ and $\mathrm{src}'=\mathrm{src}$. A particular instance is delete_a, $a\in L$, where $r=\{(l,l)\mid l\in L\setminus\{a\}\}$. (Note that relab_r is a finite composition of operations delete_a, $\mathrm{create}_{b,c,1\ldots\rho(b)}$ (introduced below) and relab_f, where f is a mapping.)
- $\mathrm{create}_{a,b,i_1\ldots i_k}$ is a unary operation of type nn, where $a,b\in L$, $k=\rho(b)\geqslant\rho(a)$ and $\{i_1,\ldots,i_k\}=[\rho(b)]$. If $G=(V,E,\mathrm{src})$ is a simple graph, $\mathrm{create}_{a,b,i_1\ldots i_k}(G)$ denotes the simple graph $G'=(V',E',\mathrm{src}')$ with b-labelled edges added in all places where they touch an a-labelled edge in positions $i_1,\ldots,i_k$ relative to a: $V'=V$, $E'=E\cup\{(b,v_{i_1}\ldots v_{i_k})\mid(a,v_1\ldots v_k)\in E\}$ and $\mathrm{src}'=\mathrm{src}$.
- $\mathrm{add}_{a,b,c}$ is a unary operation of type nn, where $a,b,c\in L$ and $\rho(c)=\rho(a)+\rho(b)$. If $G=(V,E,\mathrm{src})$ is a simple graph, $\mathrm{add}_{a,b,c}(G)$ denotes the simple graph $G'=(V',E',\mathrm{src}')$ with c-labelled edges added in all places where their first part lies parallel to an a-labelled edge and their second to a b-labelled edge: $V'=V$, $E'=E\cup\{(c,v_1\ldots v_{\rho(c)})\mid(a,v_1\ldots v_{\rho(a)}),(b,v_{\rho(a)+1}\ldots v_{\rho(c)})\in E\}$ and $\mathrm{src}'=\mathrm{src}$.
- keep_a is a unary operation of type mn, where $a\in L$, $\rho(a)=1$. If $G=(V,E,\mathrm{src})$ is a simple graph, $\mathrm{keep}_a(G)$ denotes the full subgraph $G'=(V',E',\mathrm{src}')$ of G in which all non-source vertices are adjacent to an a-labelled edge: $V'=\{v\in V\mid(a,v)\in E\}\cup\{\mathrm{src}(i)\mid i\in[n]\}$, $E'=E\cap L\times V'^*$ and $\mathrm{src}'=\mathrm{src}$.

The HR operations ren_f and fuse_δ as well as all VR operations can be defined accordingly for arbitrary finite relational structures.

6.2. Definable transductions

We extend the vocabularies introduced above to languages which are able to express properties of relational structures. *Monadic second-order logic* (*MSOL*) is the extension of (first-order) predicate logic with equality that allows quantification for unary predicates (sets of individuals) besides quantification for individuals. *Counting monadic second-order logic* (*CMSOL*) contains additional "predicates" $\mathrm{card}_{p,q}$, where $\mathrm{card}_{p,q}(X)$ holds if $|X| \equiv p \pmod q$. (X is a unary predicate, hence a set.) The *quantification depth* of a formula is the maximum number of nested quantifiers. Let $\vec{x}\vec{X}$ be a finite sequence (without repetitions) of *object variables* (lowercase) and/or *set variables* (uppercase). We denote by $\mathfrak{L}(\mathscr{R},\mathscr{C},\vec{x}\vec{X})$ the set of CMSOL formulas written with predicate symbols $\mathscr{R}$, constants $\mathscr{C}$ and free variables in $\vec{x}\vec{X}$. (Other variables may well be used in quantifications.) Let $\mathfrak{L}^{d,r}(\mathscr{R},\mathscr{C},\vec{x}\vec{X})$ be the set of such formulas with quantification depth at most d and modulus q (in $\mathrm{card}_{p,q}$) at most r.

Remark 6.13. Equality does not increase the expressive power of full MSOL, because $x = y \iff \forall P\,(P(x) \leftrightarrow P(y))$. But its inclusion is nevertheless important for measuring the quantification depth of a formula.

There are only finitely many different formulas up to tautological equivalence in $\mathfrak{L}^{d,r}(\mathscr{R},\mathscr{C},\vec{x})$. Therefore, one can define a locally finite set of predicates $\mathscr{P}^{d,r}_{\mathscr{R},\mathscr{C}}$ on $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ in a very intuitive sense: The set $\mathscr{P}_{(\mathscr{R},C)}$ of predicates of sort $(\mathscr{R},C)$, $C \subseteq \mathscr{C}$, is $\mathfrak{L}^{d,r}(\mathscr{R},C,\varepsilon)$ up to tautological equivalence. For $\psi \in \mathscr{P}_{(\mathscr{R},C)}$,

$$\hat{\psi} = \{S \in \mathfrak{S}(\mathscr{R},C) \mid S \models \psi\}$$

is the class of structures *defined* by ψ.

The following theorem has been proved in [4, 6].

Fact 6.14. *The set of predicates* $\mathscr{P}^{d,r}_{\mathscr{R},\mathscr{C}}$ *is* $\mathscr{F}_{\mathscr{R},\mathscr{C}}$*-inductive.*

It follows that every definable class of finite relational structures is deterministically recognizable.

Now we turn to logical descriptions of transformations between finite relational structures. As announced earlier, we will drop the constants and represent them by unary relations instead.

Definition 6.15 (*definable transduction*). Let $(\mathscr{R},\mathscr{C})$, $(\mathscr{R}',\mathscr{C}')$ be two signatures. Let $X_1 \ldots X_n$ be a finite sequence (without repetitions) of (object or) set variables, called *parameters*. Let $k \geqslant 1$. A *definition scheme* Δ from $\mathscr{R} \cup \mathscr{C}$ to $\mathscr{R}' \cup \mathscr{C}'$ consists of:

- a *domain formula* $\delta \in \mathfrak{L}(\mathscr{R} \cup \mathscr{C}, \emptyset, X_1 \ldots X_n)$,
- k formulas $\psi_i \in \mathfrak{L}(\mathscr{R} \cup \mathscr{C}, \emptyset, X_1 \ldots X_n x_1)$, $i \in [k]$, selecting the elements in each copy, and
- a family of formulas $\theta_{r,\vec{j}} \in \mathfrak{L}(\mathscr{R} \cup \mathscr{C}, \emptyset, X_1 \ldots X_n x_1 \ldots x_{\alpha(r)})$, $r \in \mathscr{R}' \cup \mathscr{C}'$, $\vec{j} \in [k]^{\alpha(r)}$, determining part of a relation for a given combination of copies.

Let $S \in \mathfrak{S}(\mathscr{R} \cup \mathscr{C}, \emptyset)$, let $Y_1, \ldots, Y_n \subseteq D_S$. A structure T with domain $D_T \subseteq D_S \times [k]$ is *defined by* Δ *in* S if $S \models \delta(Y_1, \ldots, Y_n)$. If this condition is satisfied, T is determined as follows:

- $D_T = \{(d,i) \in D_S \times [k] | S \models \psi_i(Y_1, \ldots, Y_n, d)\}$,
- $r_T((d_1,i_1),\ldots,(d_a,i_a)), a = \alpha(r)$, holds if $(d_1,i_1),\ldots,(d_a,i_a) \in D_T$ and $S \models \theta_{r,i_1\ldots i_a}(Y_1,\ldots,Y_n,d_1,\ldots,d_a)$.

A transduction $\bar{\varphi}: \mathfrak{S}(\mathscr{R} \cup \mathscr{C}, \emptyset) \to \mathfrak{P}(\mathfrak{S}(\mathscr{R}' \cup \mathscr{C}', \emptyset))$ is *definable* if there exists a definition scheme Δ from $\mathscr{R} \cup \mathscr{C}$ to $\mathscr{R}' \cup \mathscr{C}'$ such that $\bar{\varphi}(S)$ is the set of all structures defined by Δ in S (up to isomorphism).

A definable transduction with $k = 1$ is called *noncopying*.

Example 6.16 (*definable transduction*). There is an obvious translation of simple graphs $G = (V, E, \mathrm{src})$ into graphs $G' = (V, E, \mathrm{vert}, \mathrm{lab}, \mathrm{src})$ of the same sort: $\mathrm{vert}((l, \vec{v})) = \vec{v}$ and $\mathrm{lab}((l, \vec{v})) = l$. The converse is a parameterless, noncopying definable transduction between the associated relational structures:

- $\delta \leftrightarrow \mathsf{true}$.
- Throw away the edges: $\psi_1 \leftrightarrow \neg \bigvee_{l \in L} \exists v_1 \ldots \exists v_{\rho(l)} \mathrm{edge}_l(x_1, v_1, \ldots, v_{\rho(l)})$.
- Simplify the edge relations: $\theta_{\mathrm{edge}'_l,1\ldots1} \leftrightarrow \exists e\, \mathrm{edge}_l(e, x_1, \ldots, x_{\rho(l)})$ for every $l \in L$.
- Keep the sources: $\theta_{s_i,1} \leftrightarrow s_i(x_1)$ for every $i \in [n]$.

It is not necessary to impose additional constraints in this example. By assumption, $\theta_{s_i,1}$ is satisfied by exactly one element. In general, such constraints would go into δ, using the formulas $\theta_{r,\vec{j}}$.

Definable transductions possess similar closure properties as sets of evaluations which are compatible with a transducer between functional magmas: They are closed under composition and union, they contain intersections with definable sets, and the inverse image of a definable transduction is definable. See [7].

6.3. Logical transformations are algebraic

We show that every definable transduction is compatible with a transducer. This proves directly that images of equational sets under definable transductions are equational. Proposition 5.6 allows to compute a system of equations for the image. This improves the achievements of [13, 6], which prove the same fact but have to reuse the original derivation trees with different interpretations.

The following fact has been proved in [7]. In the definable transduction copy_k we concede more freedom to the specification of constants than there was present in [6].

Fact 6.17. *Every definable transduction can be written as a composition of two definable transductions* $\varphi \circ \mathrm{copy}_k$, *where* φ *is noncopying and* copy_k *is a definable transduction specified as follows. Let* $S \in \mathfrak{S}(\mathscr{R}, \mathscr{C})$. *Then* $\mathrm{copy}_k(S)$ *is the following structure* $T \in \mathfrak{S}(\bar{\mathscr{R}}, \bar{\mathscr{C}})$, $\bar{\mathscr{R}} = \mathscr{R} \cup \{\mathrm{son}_i \mid i \in [k]\} \cup \{\mathrm{brother}\}$, $\bar{\mathscr{C}} = \mathscr{C} \times [k]$:

- $D_T = D_S \times [k]$.
- $r_T = \{((d_1,i),\ldots,(d_a,i)) \mid a = \alpha(r), (d_1,\ldots,d_a) \in r_S, i \in [k]\}$ *for every* $r \in \mathscr{R}$.

- $\mathrm{son}_{iT} = \{(d,i) \mid d \in D_S\}$ *for every* $i \in [k]$.
- $\mathrm{brother}_T = \{((d,i),(d,j)) \mid d \in D_S, i,j \in [k]\}$.
- $(c,i)_T = (c_S,i)$ *for every* $c \in \mathscr{C}$ *and* $i \in [k]$. (*Note that* $(c,i)_T$ *is the unique element satisfying the unary relations* c_T *and* son_{iT} *at the same time.*)

We show that $\mathrm{copy}_k(S)$ is a regular image of S. Here and in the following we overload the names of evaluations and use the same symbol for every member of a set.

Lemma 6.18 (copy_k). *The set of evaluations* copy_k *on* $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ *in* $\mathscr{M}_{\bar{\mathscr{R}},\bar{\mathscr{C}}}$ *is compatible with a deterministic transducer* (*and* $\mathscr{F}_{\mathscr{R},\mathscr{C}}$*-inductive*).

Proof. We have

- $\mathrm{copy}_k(0_{\mathscr{R},\emptyset}) = 0_{\bar{\mathscr{R}},\emptyset}$.
- $\mathrm{copy}_k(1_{\mathscr{R},C}) = K_C$, $C \subseteq \mathscr{C}$, where K_C, the "clique" with k "vertices", is determined as follows: $D_{K_C} = [k]$, $r_{K_C} = \{(i,\ldots,i) \mid i \in [k]\}$ (the sequence of i's has length $\alpha(r)$) for every $r \in \mathscr{R}$, $\mathrm{son}_{iK_C} = \{i\}$ for every $i \in [k]$, $\mathrm{brother}_{K_C} = [k] \times [k]$, $(c,i)_{K_C} = i$ for every $c \in C$. This structure can be written with the operations in $\mathscr{F}_{\bar{\mathscr{R}},\bar{\mathscr{C}}}$. (Sufficient are, for example, $1_{\bar{\mathscr{R}},C \times \{i\}}$, $i \in [k]$ and // together with the VR operations delete_r, $r \in \{\mathrm{son}_i \mid i \in [k]\} \cup \{\mathrm{brother}\}$, and $\mathrm{add}_{\mathrm{son}_i,\mathrm{son}_j,\mathrm{brother}}$, $i,j \in [k]$, $i \neq j$.)
- Let a quantifier-free definition scheme Δ from $(\mathscr{R},C)$ to $(\mathscr{R},C')$, $C,C' \subseteq \mathscr{C}$, be specified by ψ, θ_r, $r \in \mathscr{R}$, and κ_c, $c \in C'$. We specify a corresponding $\bar{\Delta}$ on the larger structure:

$$\bar{\theta}_r \leftrightarrow \bigvee_{i \in [k]} \left(\theta_r[c := (c,i) \mid c \in C] \wedge \mathrm{son}_i(x_1) \wedge \ldots \wedge \mathrm{son}_i(x_{\alpha(r)})\right),$$

  $r \in \mathscr{R}$, and similarly for $\bar{\psi}$. Moreover, $\bar{\theta}_{\mathrm{son}_i} \leftrightarrow \mathrm{son}_i(x_1)$, $i \in [k]$, and $\bar{\theta}_{\mathrm{brother}} \leftrightarrow \mathrm{brother}(x_1,x_2)$. Finally, $\bar{\kappa}_{(c,i)} = (\kappa_c,i)$, $c \in C'$. Then, $\mathrm{copy}_k(\mathrm{def}_\Delta(S)) = \mathrm{def}_{\bar{\Delta}}(\mathrm{copy}_k(S))$.
- $\mathrm{copy}_k(S_1 // S_2) = \mathrm{copy}_k(S_1) // \mathrm{copy}_k(S_2)$.

It is easy to turn these second-order substitutions into a deterministic transducer with one state. □

The following observation allows to eliminate the parameters of a definable transduction: Every definable transduction can be written as a composition of two definable transductions $\varphi \circ \mathrm{param}$, where φ is parameterless. We simply include the parameters in an intermediate structure as unary relations, so that φ can use the original definition scheme. Every possible choice of parameters gives rise to a different intermediate structure. Let $S \in \mathfrak{S}(\mathscr{R},\mathscr{C})$. Then $\mathrm{param}(S)$ is a set of structures in $\mathfrak{S}(\bar{\mathscr{R}},\bar{\mathscr{C}})$, $\bar{\mathscr{R}} = \mathscr{R} \cup \{X_1,\ldots,X_n\}$, $\bar{\mathscr{C}} = \mathscr{C}$. All elements of $\mathrm{param}(S)$ contain S, that is, they consist of D_S, r_S ($r \in \mathscr{R}$) and c_S ($c \in \mathscr{C}$) besides $X_1,\ldots,X_n$.

We show that $\mathrm{param}(S)$ is a regular image of S.

Lemma 6.19 (param). *The set of evaluations* param *on* $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ *in* $\mathscr{M}_{\bar{\mathscr{R}},\bar{\mathscr{C}}}$ *is compatible with a transducer* (*and* $\mathscr{F}_{\mathscr{R},\mathscr{C}}$*-inductive*).

Proof. We have

- $\mathrm{param}(0_{\mathscr{R},\emptyset}) = \{0_{\bar{\mathscr{R}},\emptyset}\}$.
- $\mathrm{param}(1_{\mathscr{R}}, C) = \{\mathrm{Delete}_{\mathscr{X}}(1_{\bar{\mathscr{R}},C}) \mid \mathscr{X} \subseteq \{X_1,\ldots,X_n\}\}$, $C \subseteq \mathscr{C}$, where $\mathrm{Delete}_{\mathscr{X}}$ is the finite composition of delete_X operations, $X \in \mathscr{X}$.
- Let a quantifier-free definition scheme Δ from $(\mathscr{R}, C)$ to $(\mathscr{R}, C')$, $C, C' \subseteq \mathscr{C}$, be specified by ψ, θ_r, $r \in \mathscr{R}$, and κ_c, $c \in C'$. We specify a corresponding $\bar{\Delta}$ on the larger structure: $\bar{\psi} \leftrightarrow \psi$, $\bar{\theta}_r \leftrightarrow \theta_r$, $r \in \mathscr{R}$, $\bar{\theta}_{X_i} \leftrightarrow X_i(x_1)$, $i \in [n]$, and $\bar{\kappa}_c = \kappa_c$, $c \in C'$. Then, $\mathrm{param}(\mathrm{def}_{\Delta}(S)) = \mathrm{def}_{\bar{\Delta}}(\mathrm{param}(S))$,
- $\mathrm{param}(S_1 // S_2) = \mathrm{param}(S_1) // \mathrm{param}(S_2)$.

It is easy to turn these finite sets of second-order substitutions into a transducer with one state. □

Theorem 6.20. *If $\bar{\varphi}: \mathfrak{S}(\mathscr{R} \cup C, \emptyset) \to \mathfrak{S}(\bar{\mathscr{R}} \cup \bar{\mathscr{C}}, \emptyset)$ is a definable transduction then there exists a finite set $\bar{L}$ such that φ belongs to a set of evaluations which is compatible with a transducer from $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ to $\mathscr{M}_{\bar{L}\cup\bar{\mathscr{R}},\bar{L}\cup\bar{\mathscr{C}}}$, whenever $C \subseteq \mathscr{C}$.*

A similar statement was proved in [13, 6] for the special case that the domain is a set of trees. The following proof uses ideas from [6, 15].

Proof. By the two preceding lemmas and closure under composition (Proposition 5.7) we can assume that φ is parameterless and noncopying, hence also functional. Let d, r, n be the maximum quantification depth, the maximum modulus q (in $\mathrm{card}_{p,q}$) and the maximum number of free object variables, respectively, with respect to the formulas δ, ψ_1 and $\theta_{R,1\ldots1}$, $R \in \bar{\mathscr{R}} \cup \bar{\mathscr{C}}$, in the definition scheme for φ. By Fact 6.14, we can build a deterministic automaton (σ, A, F) on $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ with set of states contained in $\mathfrak{P}(\mathscr{P}^{d,r}_{\mathscr{R},\mathscr{C}})$ which maps a structure to the set of (equivalence classes of) formulas it satisfies: $\sigma_{(\mathscr{R},C)}: S \mapsto \{\chi \in \mathscr{P}_{(\mathscr{R},C)} \mid S \models \chi\}$. It is able to recognize those structures satisfying δ. (Note that the set of states is not quite as large as it might seem. Those sets of formulas which are actually used contain a canonical representative, namely the conjunction of all formulas in it.) Moreover, let us write L_X, $X \subseteq \{x_1,\ldots,x_n\}$, for the subset of (equivalence classes of) formulas in $\mathscr{P}^{d,r}_{\mathscr{R},\mathscr{C}\cup X}$ which actually contain all the variables in X free. We can view every $\chi \in L_X$, $X \neq \emptyset$, as a $|X|$-ary relation: In a finite relational structure S, $\chi_S(d_1,\ldots,d_m)$ holds for $d_1,\ldots,d_m \in D_S$ if $S \models \chi(d_1,\ldots,d_m)$. D_S together with the relations χ_S and the constants greatly enhance the original structure S, because the atomic formulas still represent the original relations (and constants). We call this new structure $\mathrm{sat}_{d,r,n}(S)$.

Let $L = \bigcup_{\emptyset \neq X \subseteq \{x_1,\ldots,x_n\}} L_X$ and let L_1 and L_2 be two copies of L, both disjoint from L and each other. Let $h_1: L \to L_1$ and $h_2: L \to L_2$ be two bijections. We abbreviate $L_X \cup h_1(L_X) \cup h_2(L_X)$, $X \subseteq \{x_1,\ldots,x_n\}$, to $\bar{L}_X$ and set $\bar{L} = L \cup L_1 \cup L_2$.

Claim 6.21. *The set of evaluations $\mathrm{sat}_{d,r,n}$ on $\mathscr{M}_{\mathscr{R},\mathscr{C}}$ in $\mathscr{M}_{\bar{L},\mathscr{C}}$ is compatible with a deterministic transducer.*

We do as if L_X would contain formulas, so that we need not speak about equivalence classes.

Proof. The automaton was already sketched. It remains to give the decompositions. They once more rely on Fact 6.14:

- $\mathrm{sat}_{d,r,n}(0_{\mathscr{R},\emptyset}) = 0_{\bar{L},\emptyset}$.
- $\mathrm{sat}_{d,r,n}(1_{\mathscr{R},C})$, $C \subseteq \mathscr{C}$, is a fixed structure in $\mathfrak{S}(\bar{L}, C)$. It can be written with the operations in $\mathscr{F}_{\bar{L},\mathscr{C}}$, even with $1_{\bar{L},C}$ and the VR operations delete_χ, $\chi \in \bar{L}$, alone.
- Let a quantifier-free definition scheme Δ from $(\mathscr{R}, C)$ to $(\mathscr{R}, C')$, $C, C' \subseteq \mathscr{C}$, be specified by ψ, θ_R, $R \in \mathscr{R}$, and κ_c, $c \in C'$. We specify a rank-preserving relation $\lambda \subseteq L \times L$ such that

$$\mathrm{sat}_{d,r,n}(\mathrm{def}_\Delta(S)) = \mathrm{relab}_\lambda(\mathrm{keep}_\psi(\mathrm{ren}_\kappa(\mathrm{sat}_{d,r,n}(S)))) .$$

(We regard $\kappa : c \mapsto \kappa_c$ as a mapping from C' to C.) By Fact 6.14, for every $X \subseteq \{x_1, \ldots, x_n\}$, there is a mapping $\tau_X : L_X \to L_X$ which translates χ from $\mathrm{def}_\Delta(S)$ to S, i.e.,

$$\mathrm{def}_\Delta(S) \models \chi(d_1, \ldots, d_m) \iff S \models \tau_X(\chi)(d_1, \ldots, d_m)$$

for all $d_1, \ldots, d_m \in D_{\mathrm{def}_\Delta(S)}$, $m = |X|$. It is then clear that

$$\lambda = \bigcup_{\emptyset \neq X \subseteq \{x_1, \ldots, x_n\}} \tau_X^{-1}$$

does the trick.

- We specify a rank-preserving relation $\mu \subseteq \bar{L} \times L$ and a finite composition Add of $\mathrm{add}_{\chi_1,\chi_2,\chi_3}$ and $\mathrm{create}_{\chi_3,\chi,i_1\ldots i_k}$ operations such that

$$\mathrm{sat}_{d,r,n}(S_1 // S_2) = \mathrm{relab}_\mu(\mathrm{Add}(\mathrm{relab}_{h_1}(\mathrm{sat}_{d,r,n}(S_1)) // \mathrm{relab}_{h_2}(\mathrm{sat}_{d,r,n}(S_2)))) .$$

By Fact 6.14, for every pair $X_1, X_2 \subseteq \{x_1, \ldots, x_n\}$ such that every variable in X_1 comes before every variable in X_2 with respect to the ordering of variables, there is a mapping $\tau_{X_1,X_2} : L_{X_1 \cup X_2} \to \mathfrak{P}(L_{X_1} \times L_{X_2})$ which translates χ from $S_1 // S_2$ to S_1 and S_2, i.e.,

$S_1 // S_2 \models \chi(d_1, \ldots, d_m) \iff S_1 \models \chi_1(d_1, \ldots, d_k)$ and $S_2 \models \chi_2(d_{k+1}, \ldots, d_{k+l})$ for some $(\chi_1, \chi_2) \in \tau_{X_1,X_2}(\chi)$ for all $d_1, \ldots, d_k \in D_{S_1}$, $k = |X_1|$, and $d_{k+1}, \ldots, d_{k+l} \in D_{S_2}$, $l = |X_2|$, such that $m = k + l$. The composition Add contains exactly those operations $\mathrm{add}_{h_1(\chi_1),h_2(\chi_2),\chi_3}$ (in any order) such that $\chi_1 \in L_{X_1}$, $X_1 \neq \emptyset$, $\chi_2 \in L_{X_2}$, $X_2 \neq \emptyset$, $\chi_3 \in L_{X_1 \cup X_2}$ and $(\chi_1, \chi_2) \in \tau_{X_1,X_2}(\chi_3)$. Together with each operation $\mathrm{add}_{h_1(\chi_1),h_2(\chi_2),\chi_3}$, Add contains exactly those operations $\mathrm{create}_{\chi_3,\chi,i_1\ldots i_k}$ (in any order) such that $\chi \in L_X$, $X_1 \cup X_2 \subseteq X$, $k = |X|$ and χ_3 results from χ by the simultaneous substitution of the i_jth variable in $X_1 \cup X_2$ for the jth variable in X, $j \in [k]$. The relation μ depends on the respective root states $\Sigma_1 = \sigma_{(\mathscr{R},C_1)}(S_1) \subseteq L_\emptyset$, $\Sigma_2 = \sigma_{(\mathscr{R},C_2)}(S_2) \subseteq L_\emptyset$ of the automaton on S_1 and S_2. It contains the following pairs:

- (χ, χ), $\chi \in L_X$,
- $(h_1(\chi), \chi')$, $(\chi_1, \chi) \in \tau_{\emptyset,X}(\chi')$ for some $\chi_1 \in \Sigma_1$, $\chi, \chi' \in L_X$,
- $(h_2(\chi), \chi')$, $(\chi, \chi_2) \in \tau_{X,\emptyset}(\chi')$ for some $\chi_2 \in \Sigma_2$, $\chi, \chi' \in L_X$.

Again, the operations modify the structure according to the family of translations τ_{X_1,X_2}.

Thus the valid formulas without free variables constitute the state of the automaton, and the others are encoded in the structure itself during its construction. □

(*Continuing the theorem.*) We can identify the constant $c \in \mathscr{C}$ with the formula $x_1 = c$ in $L_{\{x_1\}}$. Then, $\mathrm{sat}_{d,r,n}$ takes its values in $\mathscr{M}_{\bar{L},\bar{L}}$. Now it would be nice to finish the construction with something like

$$\varphi(S) = relab_\lambda(\mathrm{keep}_{\psi_1}(\mathrm{ren}_f(\mathrm{sat}_{d,r,n}(S))))$$

for $S \in \mathfrak{S}(\mathscr{R}, C)$, $C \subseteq \mathscr{C}$, where the relation λ contains the pairs $(\theta_{R,1\ldots1}, R)$, $R \in \bar{\mathscr{R}}$. Nondeterminism would allow us to put these operations on top of the term $\mathrm{sat}_{d,r,n}(S)$. Unfortunately, only the relations in $\bar{\mathscr{R}}$ can be introduced in this way, but not the constants. Although, by assumption, the relation $\theta_{c,1}$ contains a unique element for every $c \in \bar{\mathscr{C}}$, it is impossible to assign it to c. The operation ren_f (like quantifier-free definable operations in general) can only rename existing constants, not create them from relations. In general, the constants in $\bar{\mathscr{C}}$ have nothing to do with the present constants in C. The unique element in $\theta_{c,1}$ need not be denoted by any constant in C. Therefore, it is impossible to specify a suitable mapping $f : \bar{\mathscr{C}} \to C$. We remedy the situation by introducing additional constants. This is the task of another transducer, which can do some cleanup besides. We can assume that $\bar{L}$, $\bar{\mathscr{R}}$ and $\bar{\mathscr{C}}$ are pairwise disjoint.

Claim 6.22. *The set of evaluations* const : $\mathrm{sat}_{d,r,n}(S) \mapsto \varphi(S)$ *on a submagma of* $\mathscr{M}_{\bar{L},\bar{L}}$ *in* $\mathscr{M}_{\bar{L}\cup\bar{\mathscr{R}},\bar{L}\cup\bar{\mathscr{C}}}$ *belongs to a locally finite,* $\mathscr{F}_{\bar{L},\bar{L}}$*-inductive set of evaluations.*

Proof. Let $\mathscr{V}_{(\bar{L},C)} = \{v_{\chi,\Upsilon} \mid \chi \in \bar{L}_{\{x_1\}}, \Upsilon \subseteq \bar{L}_{\{x_1\}}\}$ for every $C \subseteq \bar{L}$ and define $\hat{v}_{\chi,\Upsilon}(S) = \mathrm{keep}_\chi(S)$, adding the constants in Υ to S at the same time. Let us abbreviate (the equivalence class of) the formula $x_1 \neq x_1$ to **false**. We show that $\mathscr{V}$ is $\mathscr{F}_{\bar{L},\bar{L}}$-inductive:

- $\hat{v}_{\chi,\emptyset}(0_{\bar{L},\emptyset}) = \{0_{\bar{L}\cup\bar{\mathscr{R}},\emptyset}\}$ and $\hat{v}_{\chi,\Upsilon}(0_{\bar{L},\emptyset}) = \emptyset$ for $\Upsilon \neq \emptyset$.
- Let Delete be the composition of all delete_R operations, $R \in \bar{\mathscr{R}}$, in any order.

$$\hat{v}_{\chi,\Upsilon}(1_{\bar{L},C}) \begin{cases} \{0_{\bar{L}\cup\bar{\mathscr{R}},\emptyset}\} & \text{if } \chi \leftrightarrow \textbf{false} \text{ (and } C = \emptyset), \\ \{\mathrm{Delete}(1_{\bar{L}\cup\bar{\mathscr{R}},C\cup\Upsilon})\} & \text{otherwise.} \end{cases}$$

- Let a quantifier-free definition scheme Δ from $(\bar{L}, C)$ to $(\bar{L}, C')$, $C, C' \subseteq \bar{L}$, be specified by ψ, θ_R, $R \in \bar{L}$, and κ_c, $c \in C'$. We require that all constants in $C \cup C'$ have the form $x_1 = c$, $c \in \mathscr{C}$, and that $\theta_{x_1=c}$ is $\kappa_{x_1=c}$ for every $c \in C'$. Note that Δ effectively transforms the relation θ_R into the relation R for every $R \in \bar{L}$. We specify a corresponding $\bar{\Delta}$ on the target structure: $\bar{\psi} \leftrightarrow \textbf{true}$, $\bar{\theta}_R \leftrightarrow \theta_R$, $R \in \bar{L}$, and $\bar{\theta}_R$ is arbitrary for $R \in \bar{\mathscr{R}}$. Moreover, $\bar{\kappa}_c = \kappa_c$, $c \in C'$, and $\bar{\kappa}_c = \theta_c$, $c \in \Upsilon$. (Both specifications agree on $C' \cap \Upsilon$, should it be nonempty.) Therefore, with $\Upsilon' = \{\theta_v \mid v \in \Upsilon\}$,

$$\hat{v}_{\chi,\Upsilon}(\mathrm{def}_\Delta(S)) = \begin{cases} \mathrm{def}_{\bar{\Delta}}(\hat{v}_{\theta_\chi \wedge \psi,\Upsilon'}(S)) & \text{if } \textbf{false} \notin \Upsilon', \\ \emptyset & \text{otherwise.} \end{cases}$$

- $\hat{v}_{\chi,\Upsilon}(S_1 // S_2) = \bigcup_{\substack{\Upsilon_1\cup\Upsilon_2=\Upsilon \\ \Upsilon_1\cap\Upsilon_2=\emptyset}} \hat{v}_{\chi,\Upsilon_1}(S_1) // \hat{v}_{\chi,\Upsilon_2}(S_2)$. (This guesswork could be guided by an extended automaton. It would suffice to increase the maximum quantification

depth by one, so that presence of the formula $\exists x_1\, \upsilon$ in the root state would indicate an occurrence of the relation $\upsilon \in \Upsilon$ in the substructure. But the equivalent transducer in Lemma 5.5 is nondeterministic anyway.)

Note that, despite all the transformations, the following invariants are maintained: If $\hat{\upsilon}_{\chi,\Upsilon}$ is applied to S then χ is of the form $\chi' \vee \bigvee_{c\in C} x_1 = c$ and each relation υ_S, $\upsilon \in \Upsilon$, contains a unique element.

Finally, we can add const and define $\mathrm{const}(S) = \mathrm{ren}_f(\mathrm{relab}_\lambda(\upsilon_{\{\psi\},C}(S)))$. The relation λ is as above; $f : \bar{\mathscr{C}} \to \bar{L} \cup \bar{\mathscr{C}}$, $c \mapsto \theta_{c,1}$ selects the constants. □

(*Concluding the theorem.*) The assertion follows from Lemma 5.5 and closure under composition (Proposition 5.7). □

The preceding theorem is quite robust with respect to the set of operations on the domain of the definable transduction φ, $\mathfrak{S}(\mathscr{R}, C)$, as long as all operations can be derived from those in $\mathscr{F}_{\mathscr{R},\mathscr{C}}$, $C \subseteq \mathscr{C}$. Even more, only their logical properties (i.e., $\mathscr{F}_{\mathscr{R},\mathscr{C}}$-inductivity of CMSOL formulas) are exploited. In short, any set of operations will do as long as it allows to evaluate CMSOL formulas by induction on the term representation of a finite relational structure. On the other hand, the theorem is rather sensitive to the exact set of operations on the range of φ, $\mathfrak{S}(\bar{L} \cup \bar{\mathscr{R}}, \bar{L} \cup \bar{\mathscr{C}})$, because we have used a concrete set of operations to build terms for the image structures. If we wish to use a different set of operations then we have to prove that the construction we have just finished is still possible. We cannot specialize the theorem to every reasonable set of operations, but we will cover the interesting case where the image is a set of (simple) graphs.

Corollary 6.23 (simple graphs). *Theorem 6.20 holds if the image of φ is a set of simple graphs of sort $n \in \mathbb{N}$ and only the operations* ren_f, relab_r, $\mathrm{create}_{a,b,i_1\ldots i_k}$ *and* $\mathrm{add}_{a,b,c}$ *are allowed besides constants and parallel composition.*

Proof. An analysis of the proof of the theorem reveals that no other operations are actually used. (This is the reason why we were so explicit in our construction.) The operations keep_χ introduced in the first claim are eliminated in the second (leaving all other VR operations essentially unchanged). □

The preceding corollary does not reveal any new idea when specialized to definable transductions from trees to simple graphs. It essentially repeats the proof in [6] to show that every image of an equational (or, equivalently, recognizable or definable in MSOL) set of trees under a definable transduction is an equational set (of simple graphs). The following theorem, on the other hand, contains a new proof for the main result of [13], if specialized similarly. It can be summarized as reducing the main result of [13] to the already mentioned result in [6].

Theorem 6.24 (graphs). *Theorem 6.20 holds if the image of φ is a set of graphs of sort $n \in \mathbb{N}$ and only HR operations are allowed besides the constants.*

Proof. We assume that all graphs are written with the operations $\mathscr{F}$ of the preceding corollary and sketch an $\mathscr{F}$-inductive set of evaluations to rewrite them with HR operations. The basic observation is that all relations edge_l, $l \in L$, are introduced by $\text{add}_{a,b,c}$ operations (because they are at least binary and edges are different from vertices) and that, by assumption, no two different tuples in them share their first component. (Note that $\bar{\mathscr{R}} = \{\text{edge}_l \mid l \in L\}$.) But $\text{add}_{a,b,c}$ generates at most one occurrence of c only if a, b both occur at most once. Therefore, intuitively speaking, $\text{add}_{a,b,c}$ operations are rare events; they can only occur close to the "bottom" of a term, i.e., in relatively small subterms which generate few edges. A set of evaluations can keep track of all relations edge_l to be added and all relations $r_1, \ldots, r_k$ into which they could possibly break up. It needs to "remember" the $\text{add}_{a,b,c}$, $\text{create}_{a,b,i_1 \ldots i_k}$ and relab_r operations it has encountered so far. The set of all possible "memories" is finite; an evaluation itself is the "memory", of course. One can determine a sufficient number of sources to be introduced by consulting the "memory".

To be more specific, an evaluation $v_{R,A,N,E,V}$ is determined by

- a rank-preserving relation $R \subseteq (\bar{L} \cup \bar{\mathscr{R}}) \times (\bar{L} \cup \bar{\mathscr{R}})$. It contains pairs (a, edge_l) $a \in \bar{L} \cup \bar{\mathscr{R}}$, $l \in L$, such that edge_l grows out of a relation a. R is updated whenever a relab_r operation is encountered.
- $A \subseteq \bar{L} \cup \bar{\mathscr{R}}$, the set of all added edge_l relations in disguise (due to possible relabellings). It may be updated when an $\text{add}_{a,b,c}$ (or $\text{create}_{a,b,i_1 \ldots i_k}$) operation is encountered and c (or b) corresponds to some edge_l, $l \in L$, by R. We need it to determine whether such an edge has already been added, because it is possible that several $\text{add}_{a,b,c}$ or $\text{create}_{a,b,i_1 \ldots i_k}$ operations create the same edge. The contents of this set are also affected by relab_r operations.
- $N \subseteq \bar{L} \cup \bar{\mathscr{R}}$, the set of all relations assumed to be empty. Every time when we encounter an $\text{add}_{a,b,c}$ (or $\text{create}_{a,b,i_1 \ldots i_k}$) operation, we guess whether a, b (or a) are empty or not. We must remember these guesses in order to verify them later. The contents of this set are affected by $\text{add}_{a,b,c}$ (if $c \in N$ then a or b must be included as well), $\text{create}_{a,b,i_1 \ldots i_k}$ (if $b \in N$ then a must be included as well) and relab_r operations.
- $E \subseteq \{(a,i) | a \in \bar{L} \cup \bar{\mathscr{R}}, i \in [\alpha(a)]\}$, the set of all relations containing edges besides new sources. It is like V except that one component (the ith) in each of its relations is an edge, not a new source.
- $V \subseteq \bar{L} \cup \bar{\mathscr{R}}$, the set of all relations with a unique occurrence. We have to introduce a fixed ordering on $\bar{L} \cup \bar{\mathscr{R}}$ to determine the sequence of sources introduced by them. V is extended whenever an $\text{add}_{a,b,c}$ operation is encountered, c corresponds to some edge_l, $l \in L$, by R and we guess that a and b are nonempty. In this case, a may be included in E and b in V. Likewise, if a $\text{create}_{a,b,i_1 \ldots i_k}$ operation is encountered, b corresponds to some edge_l, $l \in L$, by R and we guess that a is nonempty. The elements of V are affected by $\text{add}_{a,b,c}$ (relations c may "split" into a and b), $\text{create}_{a,b,i_1 \ldots i_k}$ (relations b may be "reduced" to a) and relab_r operations.

The following invariant holds: $\hat{v}_{R,A,N,E,V}$ is constantly the empty set if some of A, N, E, V contain a common edge. We do not need to include these evaluations explicitly, however.

We introduce the temporary notation $\#V$ for the sum of the arities of the relations in V, and $\#E$ for the sum of the arities of the relations in E minus $|E|$ (because one component in each tuple does not count). An $\mathscr{F}$-inductive set of evaluations is determined as follows: (We show only a few instructive cases and leave the rest to the reader.)

- $\hat{v}_{R,A,N,E,V}(1_{\bar{L}\cup\bar{\mathscr{R}},[1]}) = \emptyset$ if $E \neq \emptyset$ and $V \neq \emptyset$ or if E contains a relation of arity greater than one. Both conditions indicate that this constant denotes an edge and a vertex at the same time, which is impossible. Also, $\hat{v}_{R,A,N,E,V}(1_{\bar{L}\cup\bar{\mathscr{R}},[1]}) = \emptyset$ if $N \neq \emptyset$, because the relations in N were assumed to be empty. $\hat{v}_{R,A,N,E,V}(1_{\bar{L}\cup\bar{\mathscr{R}},[1]}) = \{0_{L,\emptyset}\}$ if $E \neq \emptyset$ (and all relations in it have arity one). Obviously, this constant denotes an edge and is therefore superfluous. Otherwise, $\hat{v}_{R,A,N,E,V}(1_{\bar{L}\cup\bar{\mathscr{R}},[1]}) = \{1_{L,[\#V+1]}\}$, because each component of V requires a number of new sources according to its arity.
- The case of an $\mathrm{add}_{a,b,c}$ operation is the most interesting. Let us assume that $c\mathrm{Redge}_l$ and that $l \in L$ is the unique such element. (In the general case, several or no edges may be added.) We have to guess whether the operation actually adds an edge, in which case we do it with HR operations:

$$\begin{aligned}\hat{v}_{R,A,N,E,V}(\mathrm{add}_{a,b,c}(S)) = \mathrm{ren}_f(\mathrm{fuse}_\delta(l \oplus \hat{v}_{R,A\cup\{c\},N,E',V'}(S)))\\ \cup\ \hat{v}_{R,A,N\cup\{a\},E,V}(S)\\ \cup\ \hat{v}_{R,A,N\cup\{b\},E,V}(S)\\ \cup\ \hat{v}_{R,A,N\cup\{a,b\},E,V}(S)\end{aligned}$$

and $c \notin A$. $E' = (E \setminus \{(c,1)\}) \cup \{(a,1)\}$ and $V' = V \cup \{b\}$. The number of sources $(\#E+\#V+n$, $\alpha(a)-1+\#E+\#V+n$, $\alpha(b)+\#E+\#V+n$ or $\rho(l)+\#E+\#V+n$ according to whether $(c,1) \in E$, $(a,1) \in E$, $b \in V$, where n is the sort of the graph $S)$ of $\hat{v}_{R,A\cup\{c\},N,E',V'}(S)$ is known, so an equivalence relation δ and a mapping f can be specified appropriately. If $c \in A$ then $\hat{v}_{R,A,N,E,V}(\mathrm{add}_{a,b,c}(S)) = \hat{v}_{R,A,N,E,V}(S)$.

We note that there are also a few other cases to consider, in particular that $c\mathrm{R}\,\mathrm{edge}_l$ for no $l \in L$, but $c \notin A$ and $(c,j) \in E$ for some $j \in [\alpha(c)]$ (or $c \in V$, which is similar).

- A relab_r operation mostly involves some book-keeping. Let us write $V'rV$ if both sets have the same cardinality and r holds elementwise according to the ordering on $\bar{L} \cup \bar{\mathscr{R}}$. Similarly for E' and E. Let N' be the coimage of N under r, $N' = \{a' \in \bar{L} \cup \bar{\mathscr{R}} \mid (\exists a \in N)\, a'ra\}$. Similarly for A' and A. Finally, $R' = R \circ r$. We have

$$\hat{v}_{R,A,N,E,V}(\mathrm{relab}_r(S)) = \bigcup_{\substack{E'rE\\V'rV}} \hat{v}_{R',A',N',E',V'}(S)\,.$$

The result may be empty due to a wrong guess.

- A parallel composition // requires to guess how to distribute the sources.

$$\hat{v}_{R,A,N,E,V}(S_1 // S_2) = \bigcup_{\substack{E_1\cup E_2=E, E_1\cap E_2=\emptyset\\ V_1\cup V_2=V, V_1\cap V_2=\emptyset}} \mathrm{ren}_{f_{E_1,E_2,V_1,V_2}}(\mathrm{fuse}_\delta(\hat{v}_{R,A,N,E_1,V_1}(S_1)\oplus\hat{v}_{R,A,N,E_2,V_2}(S_2)))\,.$$

The mapping f_{E_1,E_2,V_1,V_2} takes care of the distribution of sources. The equivalence relation δ, on the other hand, does not depend on the distribution; it can be read from the type of //

The image of φ is given by $v_{\{(\mathrm{edge}_i,\mathrm{edge}_i)\mid l\in L\},\emptyset,\emptyset,\emptyset,\emptyset}$. □

We do not know under which restrictions a converse of Theorem 6.20 also holds: When can a regular image of a set Q of finite relational structures be obtained through a definable transduction? A sufficient condition would be that Q is equational and there exists a definable transduction assigning to a finite relational structure in Q a derivation tree. This constitutes the first step, namely constructing a term denoting a structure. Simulating the (tree) transducer by a logical formula is routine. The last step, evaluating the obtained term to a finite relational structure, is a definable transduction by a result in [6]. However, constructing derivation trees via definable transductions seems to be difficult. A system of equations for which this is possible is called *parsable*. Courcelle conjectured in [5] that whenever a set Q of graphs is equational and definable in CMSOL then a parsable system of equations for Q can be chosen. He proved this conjecture for special cases only [5, 11]. We add as a further open question whether the analogous statement holds for a set Q of simple graphs.

7. Finite relational structures and nondeterministic operations

Now we look at examples for nondeterministic operations. Of course, our goal remains to describe sets of finite relational structures by systems of equations and prove properties of them. We cannot hope, however, that sets of formulas are inductive in the sense of the previous section, which means that the defined sets of finite relational structures would be deterministically recognizable. In general, logical formulas distinguish too fine. We confine ourselves to $\exists$-inductive and $\forall$-inductive sets of formulas. (If the logical language allows negation, like CMSOL, both notions coincide.) Even then, by Lemma 5.13, satisfiability and validity of formulas would be decidable in equational sets. At the time of this writing it is known that a set of graphs (over a finite set of labels L) having a decidable CMSOL theory is contained in an equational set (of graphs) with respect to deterministic operations [28]. Courcelle conjectured in [9] that the analogous statement holds for sets of simple graphs. (Note that CMSOL on simple graphs lacks the possibility to quantify over edges or sets of edges.) See [9] for a summary of known results. If the conjecture turns out to be true, nondeterministic operations could not be used to describe new interesting sets of finite relational structures. Their main use would be to describe an equational set by a shorter system of equations.

Lemma 7.1 (union operation). *If the set of predicates $\mathscr{P}$ for a functional magma M is inductive and $f_1,\ldots,f_k \in \mathscr{F}$ have the same type $s_1\ldots s_n s$ then $\mathscr{P}$ is $\exists$-inductive and $\forall$-inductive with respect to the relational mapping $M(e)$, $e \in \mathscr{E}_s(\mathscr{F},x_1,\ldots,x_n)$, defined by $e = f_1x_1\ldots x_n +_s \cdots +_s f_kx_1\ldots x_n$.*

Proof. Obvious. □

Union operations always introduce bounded nondeterminism because

$$|M(e)(\{m_1\},\ldots,\{m_n\})| \leqslant k$$

for all $m_i \in M_{s_i}$, $i \in [n]$. Compositions of union operations are again union operations.

Example 7.2. A combination of $|\mathscr{C}|!$ unary quantifier-free definable operations on $\mathfrak{S}(\mathscr{R},\mathscr{C})$ is *permutation of constants*. The result $\pi(S)$ of $S = (D_S, \langle r_S\rangle_{r\in\mathscr{R}}, \langle c_S\rangle_{c\in\mathscr{C}})$ is $(D_S, \langle r_S\rangle_{r\in\mathscr{R}}, \langle f(c)_S\rangle_{c\in\mathscr{C}})$ with an arbitrary permutation f of $\mathscr{C}$.

Permutation of constants can be used, e.g., to define a *randomized parallel composition* as $\pi(\pi(S_1)/\!/\,\pi(S_2))$. Although the constants are indistinguishable it is still important to have (a finite number of) names for them – see below. This operation could be used in item 4 of Example 4.6. The trees in T_i possess $\frac{5\cdot 3^i-1}{2}$ sources; one of them is chosen in step $i+1$.

Now we turn to a more general concept. We extend the signature by constants $c_1,\ldots,c_n$ and use them in a system of equations E. In general, the solution of E will depend on the values $C_i \subseteq M_{\rho(c_i)}$, $i \in [n]$, assigned to these constants. ($c_1,\ldots,c_n$ can be seen as *parameters* of E, that is, additional variables without defining equations.) We assume that the mapping $f\colon (C_1,\ldots,C_n) \mapsto \mathscr{L}(E,M).x_1$ is relational. (This will imply in general that each term appearing in E is linear, but the requirement $f(\ldots,\emptyset,\ldots) = \emptyset$ is more serious. For example, at most one constant c_i may appear in each equation if such an equation is not superfluous anyway.) Here we call f *equational.*

Lemma 7.3 (equational operation). *Assume that the set of predicates $\mathscr{P}$ for a functional magma M is inductive and that the mapping f is equational. Then $\mathscr{P}$ is $\exists$-inductive and $\forall$-inductive with respect to f.*

Proof. We can replace each constant c_i in E by an expression $c_{i,p_1} + \ldots + c_{i,p_{k_i}}$, where $p_1,\ldots,p_{k_i} \in \mathscr{P}$ are the predicates of appropriate sort and $c_{i,p}$ is interpreted by $C_i \cap \hat{p}$. By Corollary 5.11, for each $p \in \mathscr{P}$, there is a system of equations E' describing $\mathscr{L}(E,M).x_1 \cap \hat{p}$. In fact, the proof of Proposition 5.6 shows that the same E' can be chosen for each p. It contains variables of the form $[x,p]$ where x is a variable of E and $p \in \mathscr{P}$. If an equation for x in E contains the constant c_i then the equation for $[x,p]$ in E' contains the constant $c_{i,p}$ and no other constants associated with c_i. One can tell whether $f(C_1,\ldots,C_n) \cap \hat{p} = \mathscr{L}(E',M).[x_1,p] \neq \emptyset$ by determining which constants have a nonempty interpretation. This gives a decomposition of p relative to f and proves $\exists$-inductivity. $\forall$-inductivity is proved similarly. □

If we set $M = \mathscr{M}_{\mathscr{R},\mathscr{C}}$ and $\mathscr{P} = \mathscr{P}^{d,r}_{\mathscr{R},\mathscr{C}}$ in the preceding lemma then it follows from Fact 6.14 that CMSOL formulas are $\exists$-inductive and $\forall$-inductive with respect to

permutation of constants, randomized parallel composition, union operations and equational operations in general. Therefore, satisfiability and validity of a CMSOL formula can be decided relative to an equational set written with such operations and those from the preceding section. A system of equations for the defined subset must, however, start from the (probably much longer) system of equations written with deterministic operations.

In the rest of this section we give some examples to indicate why it is difficult to get beyond "equational" nondeterminism. We only consider *ordinary* simple graphs in our counterexamples, that is, $L = \{l\}$, $\rho(l) = 2$. If there is an edge from x to y then y is called a *successor* of x and x, y are *neighbours*. Let us exclude the empty graph in order to avoid irrelevant exceptions.

We first try to generalize quantifier-free definable operations to noncopying definable transductions such that all parameters are object variables and all formulas are quantifier-free. (It is out of question to allow copying transductions. We would arrive at propagating graph-0L systems [16]. The set of square grids, which has an undecidable MSOL theory, can be described by a system of equations using such operations.) The domain formula could be used to specify that the parameters are the constants, for example. But it can as well allow to create constants out of thin air, thereby producing an unbounded number of new structures. We let the operation guess_n, $n \geqslant 1$, introduce n sources at random in its argument.

We can derive a binary gluing operation $//_n$, $n \geqslant 1$, as $G_1 //_n G_2 = \mathrm{guess}_n(G_1) // \mathrm{guess}_n(G_2)$. For example, $//_1$ is the operation glue in item 2 of Example 4.6. It obviously preserves connectivity, planarity, being a tree and many other properties. Nevertheless, this is only coincidence.

Lemma 7.4. *Not even first-order (predicate logic) formulas are* $\exists$*-inductive or* $\forall$*-inductive with respect to* $//_1$.

Proof. Consider the condition that every vertex has k successors (neighbours). One simple graph G in $G_1 //_1 G_2$ satisfies it if G_1 and G_2 satisfy it *with possibly one exception*. The latter property is more complex (in terms of quantification depth or similar measures) than the property of G. Therefore, the property in question does not belong to an $\exists$-inductive set of predicates. Its negation does not belong to a $\forall$-inductive set of predicates.

Surprisingly, negation is not the problem – it can be handled by switching between $\exists$-recognizability and $\forall$-recognizability. (A proof would proceed by simultaneous induction on the structure of a formula χ.) The problem are the universal quantifiers for $\exists$-inductivity and the existential quantifiers for $\forall$-inductivity, not even the second-order ones.

The operations $//_n$, $n \geqslant 2$, are even worse. The equation

$$x = x \oplus 1 + x //_2 e + 1$$

describes all simple graphs, where the constant 1 denotes a singleton set of a vertex and the constant e denotes a singleton set of an edge (with two endpoints). The first-order theory of finite simple graphs is well known to be undecidable.

It does not help to restrict the introduction of sources to some "small" area. Take, for example, the operation that glues two arbitrarily selected neighbours. Again, not even first-order formulas are $\exists$-inductive or $\forall$-inductive with respect to gluing of neighbours. Consider the condition that every vertex has at most k successors (neighbours) different from itself and proceed as in the preceding lemma.

Since random introductions of sources are not useful, we look at nondeterministic changes of edges. A natural generalization of quantifier-free definable operations in this direction is the following: Let us extend the definition by allowing several formulas θ_l for the same $l \in L$. One of them is randomly chosen for each selection $v_1, \ldots, v_{\rho(l)} \in V$. (A further constraint could impose upper bounds on the number of applications for some of the alternatives.)

One operation of this kind is edge addition add_l. It is defined by two formulas θ_l, namely true and $\mathrm{edge}_l(x_1, \ldots, x_{\rho(l)})$, and $\theta_\lambda \leftrightarrow \mathrm{edge}_\lambda(x_1, \ldots, x_{\rho(\lambda)})$ for every $\lambda \in L \setminus \{l\}$. It adds an arbitrary number of l-labelled edges to its argument.

Lemma 7.5. *Not even first-order formulas are $\exists$-inductive or $\forall$-inductive with respect to* add_l.

Proof. The equation

$$x = x \oplus 1 + \mathrm{add}(x) + 1$$

describes all simple graphs, where the constant 1 denotes a singleton set of a vertex and add is add_l for the unique label $l \in L$.

On the other hand, this operation does not even allow to describe more simple graphs of bounded tree-width than quantifier-free definable ones. This can be seen by a straighforward adaption of [1]. (In fact, [1] arose from this investigation.)

Dual to add_l is edge deletion delete_l. It deletes an arbitrary number of l-labelled edges in its argument. Since addition of *all* l-labelled edges is a quantifier-free definable operation, delete_l is not easier to handle.

It is clear that additional constraints cannot improve the situation. The operation add_l^1 adds at most one l-labelled edge to its argument. This means that the formula $\theta_l \leftrightarrow \mathsf{true}$ is chosen for at most one sequence $v_1, \ldots, v_{\rho(l)} \in V$. But the equation

$$x = x \oplus 1 + \mathrm{add}^1(x) + 1$$

still describes all simple graphs. It is interesting to note that add_l^1 can be derived from $\mathrm{guess}_{\rho(l)}$, fusc_δ and ren_f for suitable δ and f.

Another variation on edge addition is nondeterminism in the direction of edges. Let the operation $\eta_{l,a_1,\ldots,a_{\rho(l)}}$ (compare [14]), where $a_1, \ldots, a_{\rho(l)}$ are edge labels of rank 1 (that is, vertex labels), add an l-labelled edge to a random permutation of each sequence of $\rho(l)$ vertices carrying the labels $a_1, \ldots, a_{\rho(l)}$.

Lemma 7.6. *MSO formulas are neither ∃-inductive nor ∀-inductive with respect to* $\eta_{l,a_1,\ldots,a_{\rho(l)}}$.

Proof. This follows from the results in [9]. In particular, the equation

$$x = \eta_{l,a,b}(\mathrm{relab}_f(x) \oplus 1_b) + 1_b$$

(a, b are vertex labels, 1_a, 1_b are constants denoting a singleton set of a vertex carrying the respective label, and f maps a, b to a) describes all tournaments, which have an undecidable MSOL theory.

Finally, a similar statement holds for the operation extend in item 3 of Example 4.6, because it is proved in [9] that the class of all chordal graphs has an undecidable MSOL theory.

8. Conclusions

Are nondeterministic operations useful? In view of the preceding section we cannot expect to get equational sets with a decidable CMSOL theory that cannot be described by deterministic operations also. However, their descriptions may be significantly shorter and more natural with nondeterministic operations. It could also be interesting to consider proper fragments of MSOL which are not closed under negation. Perhaps their formulas would give good examples for ∃-inductive or ∀-inductive properties.

Nondeterministic transformations, on the other hand, definitely have their merits. The open question, under which conditions can transducers be simulated by definable transductions, is challenging.

Acknowledgements

The author likes to thank Bruno Courcelle for the initial idea, enlightening discussions and many useful suggestions to improve this article.

References

[1] K. Barthelmann, How to construct a hyperedge replacement system for a context-free set of hypergraphs, Tech. Report 7, University of Mainz, 1996, submitted for publication.

[2] M. Bauderon, B. Courcelle, Graph expressions and graph rewritings, Math. Systems Theory 20 (1987) 83–127.

[3] B. Courcelle, Graph rewriting: an algebraic and logic approach, in: J. van Leeuwen, (Ed.), Handbook of Theoretical Computer Science, vol. B, Elsevier, Amsterdam, 1990, pp. 193–242.

[4] B. Courcelle, The monadic second-order logic of graphs. I. Recognizable sets of finite graphs, Inform. and Comput. 85 (1990) 12–75.

[5] B. Courcelle, The monadic second-order logic of graphs V: on closing the gap between definability and recognizability, Theoret. Comput. Sci. 80 (1991) 153–202.
[6] B. Courcelle, The monadic second-order logic of graphs VII: Graphs as relational structures, Theoret. Comput. Sci. 101 (1992) 3–33.
[7] B. Courcelle, Monadic second-order definable graph transductions: a survey, Theoret. Comput. Sci. 126 (1994) 53–75.
[8] B. Courcelle, The monadic second-order logic of graphs VI: On several representations of graphs by relational structures, Discrete Appl. Math. 54 (1994) 117–149, Erratum in 63 (1995) 199–220.
[9] B. Courcelle, The monadic second-order logic of graphs VIII: Orientations, Ann. Pure Appl. Logic 72 (1995) 103–143.
[10] B. Courcelle, Basic notions of universal algebra for language theory and graph grammars, Theoret. Comput. Sci. 163 (1996) 1–54.
[11] B. Courcelle, The monadic second-order logic of graphs X: Linear orderings, Theoret. Comput. Sci. 160 (1996) 87–143.
[12] B. Courcelle, The expression of graph properties and graph transformations in monadic second-order logic, in: G. Rozenberg (ed.), Handbook of Graph Grammars and Computing by Graph Transformation, vol. 1, World Scientific, Singapore, 1997, pp. 313–400.
[13] B. Courcelle, J. Engelfriet, A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars, Math. Systems Theory 28 (1995) 515–552.
[14] B. Courcelle, J. Engelfriet, G. Rozenberg, Handle-rewriting hypergraph grammars, J. Comput. System Sci. 46 (1993) 218–270.
[15] B. Courcelle and M. Mosbah, Monadic second-order evaluations on tree-decomposable graphs, Theoret. Comput. Sci. 109 (1993) 49–82.
[16] K. Culik II, D. Wood, A mathematical investigation of propagating graph-0L systems, Inform. and Control 43 (1979) 50–82.
[17] F. Drewes, Semirings and tree-to-graph-to-tree transductions, in: SEGRAGRA '95, Electronic Notes in Theoretical Computer Science, vol. 2, Elsevier, Amsterdam, 1995, pp. 43–50.
[18] F. Drewes, The use of tree transducers to compute translations between graph algebras, in: Graph Grammars and Their Application to Computer Science, Lecture Notes in Computer Science, vol. 1073, Springer, Berlin, 1996, pp. 196–210.
[19] S. Eilenberg, J. B. Wright, Automata in general algebras, Inform. and Control 11 (1967) 452–470.
[20] J. Engelfriet, Bottom-up and top-down tree transformations – a comparison, Math. Systems Theory 9 (1975) 198–231.
[21] J. Engelfriet, Graph grammars and tree transducers, in: Trees in Algebra and Programming – CAAP '94, Lecture Notes in Computer Science, vol. 787, Springer, Berlin, 1994, pp. 15–36.
[22] J. Engelfriet, Context-free graph grammars, in: [27], pp. 125–213.
[23] F. Gécseg, M. Steinby, Tree Automata, Académiai Kiadó, 1984.
[24] F. Gécseg, M. Steinby, Tree languages, in: [27], pp. 1–68.
[25] A. Habel, H.-J. Kreowski, W. Vogler, Decidable boundedness problems for sets of graphs generated by hyperedge replacement, Theoret. Comput. Sci. 89 (1991) 33–62.
[26] J. Mezei, J. B. Wright, Algebraic automata and context-free sets, Inform. and Control 11 (1967) 3–29.
[27] G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages vol. 3, Springer, Berlin, 1997.
[28] D. Seese, The structure of the models of decidable monadic theories of graphs, Ann. Pure Appl. Logic 53 (1991) 169–195.
[29] A. Takahashi, S. Ueno, Y. Kajitani, Minimal acyclic forbidden minors for the family of graphs with bounded path-width, Discrete Math. 127 (1994) 293–304.
[30] J. W. Thatcher, J. B. Wright, Generalized finite automata theory with an application to a decision problem of second-order logic, Math. Systems Theory 2 (1968) 57–81.